US006278991B1

(12) **United States Patent**
Ebert

(10) **Patent No.: US 6,278,991 B1**
(45) **Date of Patent: Aug. 21, 2001**

(54) **BROWSER FOR HIERARCHICAL STRUCTURES**

(75) Inventor: **Peter Ebert**, Darmstadt (DE)

(73) Assignee: **SAP Aktiengesellschaft**, Walldorf (DE)

(*) Notice: Subject to any disclaimer, the term of this patent is extended or adjusted under 35 U.S.C. 154(b) by 0 days.

(21) Appl. No.: **09/134,502**

(22) Filed: **Aug. 14, 1998**

Related U.S. Application Data

(60) Provisional application No. 60/058,946, filed on Aug. 22, 1997.

(51) **Int. Cl.**[7] **G06F 3/14**; G06F 3/00; G06T 17/20

(52) **U.S. Cl.** **707/3**; 345/357; 345/419; 707/100

(58) **Field of Search** 707/3, 101, 102, 707/103, 104, 907; 345/419, 433, 473, 335, 339, 347, 348, 352, 353, 354, 355, 356, 357, 961, 975, 976, 977, 522; 700/17; 382/209

(56) **References Cited**

U.S. PATENT DOCUMENTS

5,339,433		8/1994	Frid-Nielson	717/5
5,491,795	*	2/1996	Beaudet et al.	395/159
5,517,605		5/1996	Wolf	707/104
5,544,302	*	8/1996	Nguyen	395/161
5,546,529	*	8/1996	Bowers et al.	395/159
5,630,042		5/1997	McIntossh et al.	345/340
5,632,022	*	5/1997	Warren et al.	345/350
5,771,379	*	6/1998	Gore, Jr.	707/101
5,815,155	*	9/1998	Wolfston, Jr.	345/357
5,877,765	*	3/1999	Dickman et al.	345/349
5,923,328	*	7/1999	Griesmer	345/357
6,040,834	*	3/2000	Jain et al.	346/356
6,052,123	*	4/2000	Lection et al.	345/419

FOREIGN PATENT DOCUMENTS

96 02889 2/1996 (WO) .

OTHER PUBLICATIONS

Kumar H P et al: "Browsing hierarchical data with multi-level dynamic queries and pruning", International Journal of Human–Computer Studies, Jan. 1997, Academic Press, UK, vol. 46, No. 1, pp. 103–124.

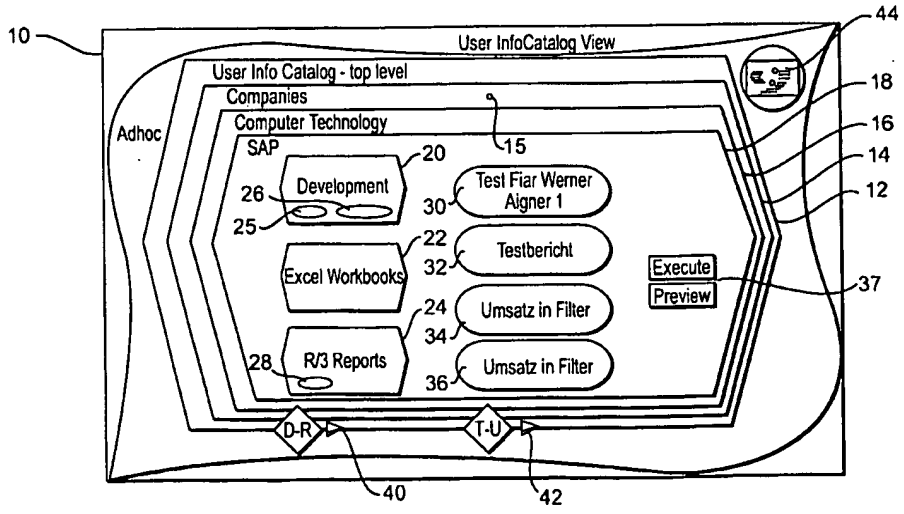
* cited by examiner

Primary Examiner—Thomas Black
Assistant Examiner—Te(Susan) Chen
(74) *Attorney, Agent, or Firm*—Finnegan, Henderson, Farabow, Garrett & Dunner L.L.P.

(57) **ABSTRACT**

A browser for viewing data representing text, pictures, sounds, virtual reality worlds, links, or other objects, and organized in a hierarchical structure is provided to display and interact with links to the data. A retrieved set of data is displayed in a lens, and subsequently retrieved sets of data that are deeper in the hierarchical structure are displayed in further stacked lenses, creating a visual representation of the hierarchy. A label on each lens provides information on the data originally displayed in the lens. A user may choose to view the originally displayed contents of any lens on the screen. A user may also modify a personal hierarchical data structure comprising links to sets of data in the first hierarchical structure. In doing so, a user creates an entirely new hierarchical structure that customizes the organization of data according to personal preferences and takes advantage of natural association to reduce the number of perceived levels in the new hierarchical data structure.

50 Claims, 13 Drawing Sheets



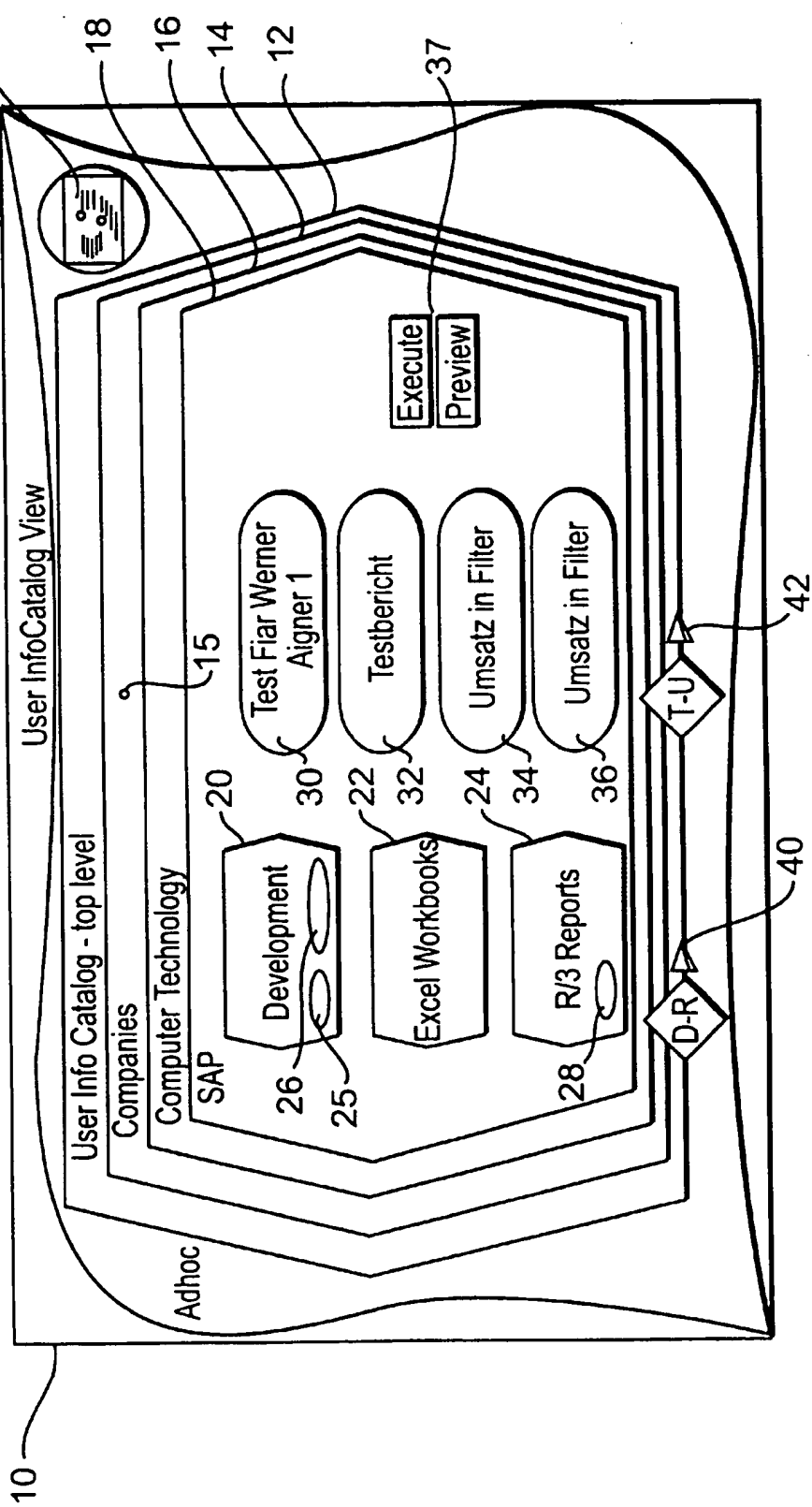

44
18
16
14
12
37
User InfoCatalog View
User Info Catalog - top level
Companies
Computer Technology
SAP
15
Execute
Preview
Test Fiar Werner Aigner 1
Testbericht
Umsatz in Filter
Umsatz in Filter
20
30
22
32
24
34
36
Development
Excel Workbooks
R/3 Reports
26
25
28
T-U
D-R
42
40
Adhoc
10

FIG. 1

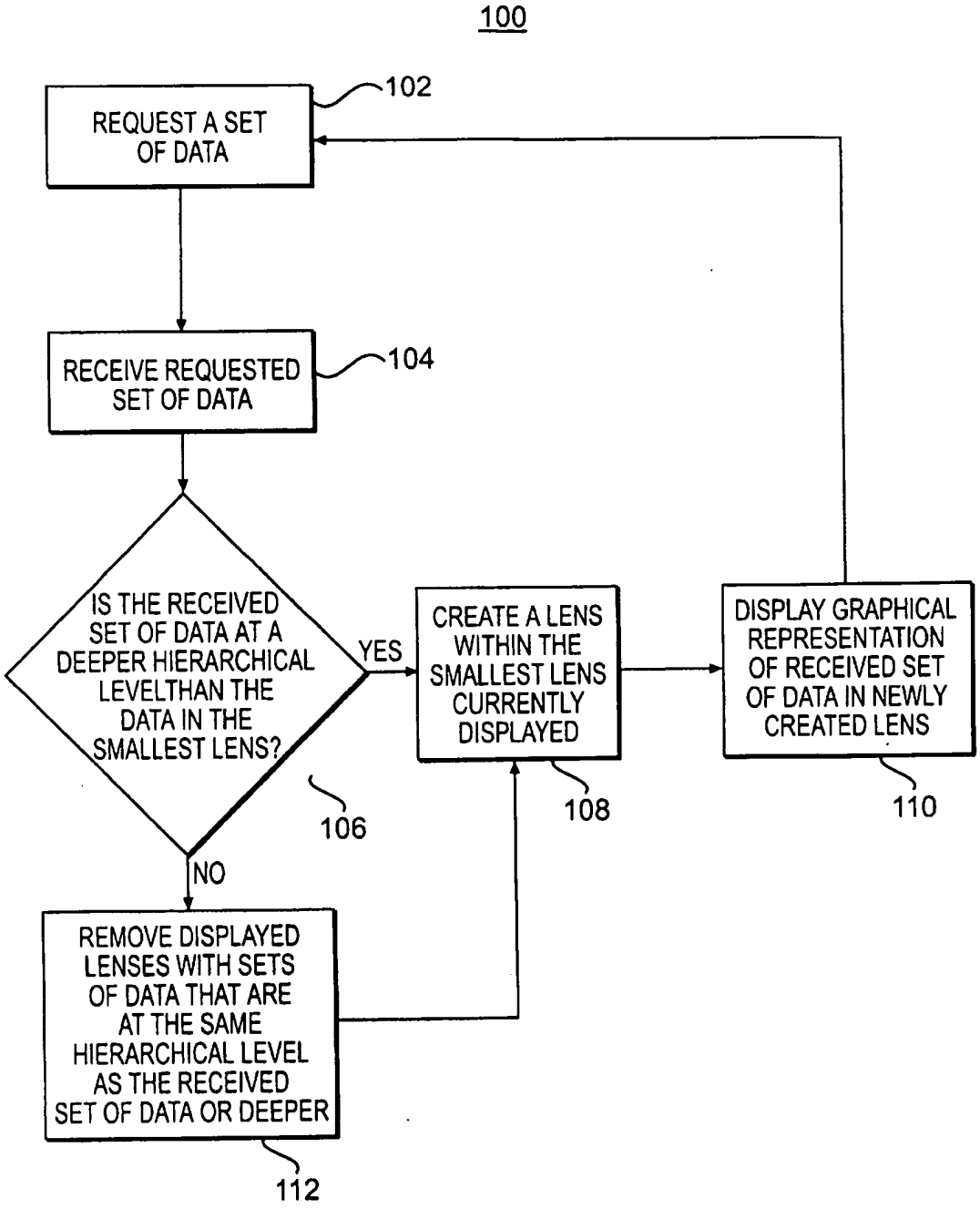

100
REQUEST A SET OF DATA
102
RECEIVE REQUESTED SET OF DATA
104
IS THE RECEIVED SET OF DATA AT A DEEPER HIERARCHICAL LEVELTHAN THE DATA IN THE SMALLEST LENS?
YES
NO
106
CREATE A LENS WITHIN THE SMALLEST LENS CURRENTLY DISPLAYED
108
DISPLAY GRAPHICAL REPRESENTATION OF RECEIVED SET OF DATA IN NEWLY CREATED LENS
110
REMOVE DISPLAYED LENSES WITH SETS OF DATA THAT ARE AT THE SAME HIERARCHICAL LEVEL AS THE RECEIVED SET OF DATA OR DEEPER
112

FIG. 2

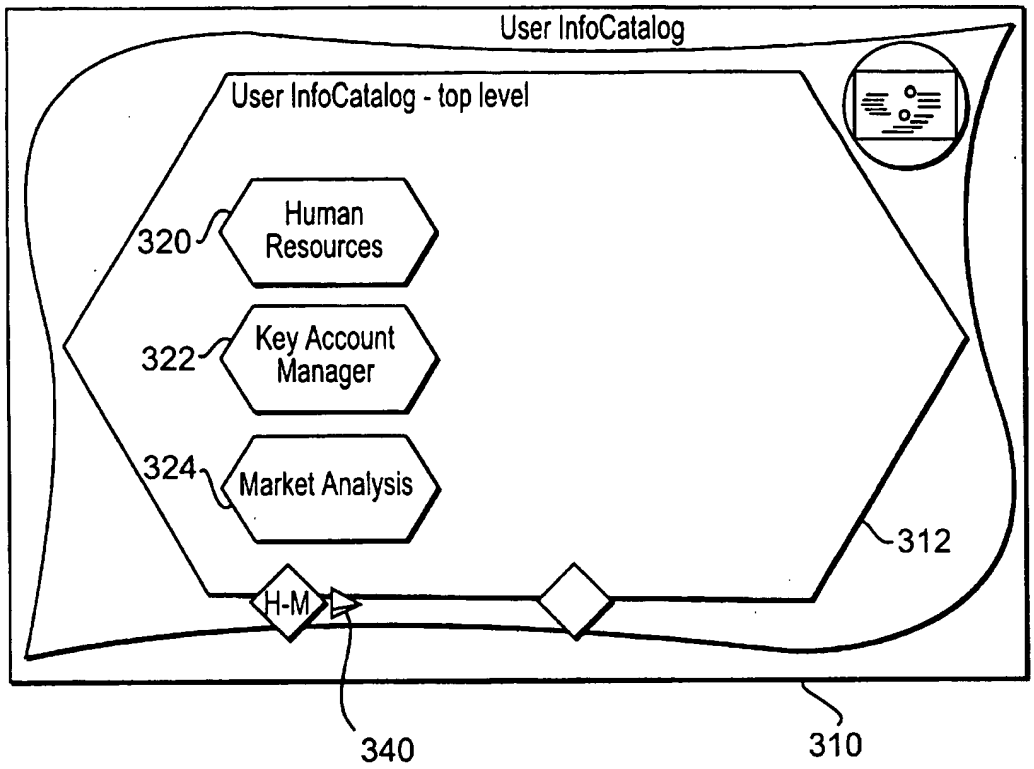

User InfoCatalog
User InfoCatalog - top level
320
Human Resources
322
Key Account Manager
324
Market Analysis
312
H-M
340
310

FIG. 3A

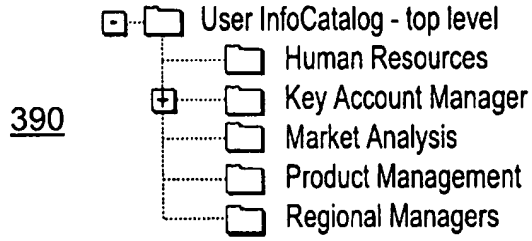

User InfoCatalog - top level
Human Resources
Key Account Manager
Market Analysis
Product Management
Regional Managers
390

FIG. 3B

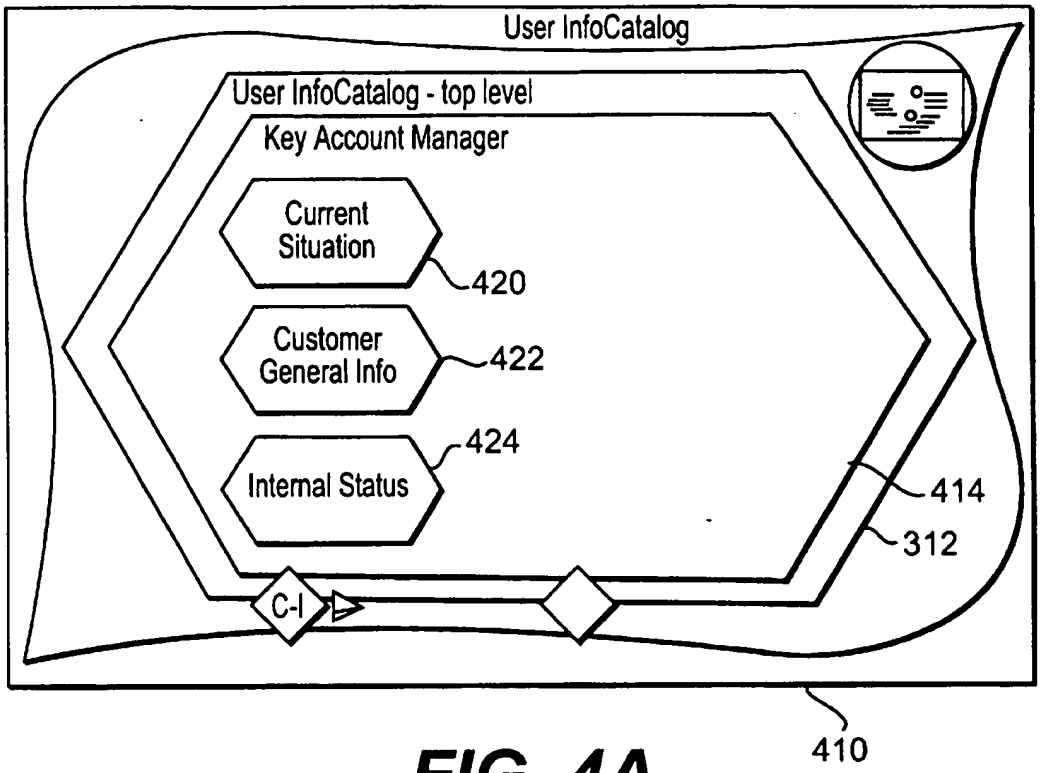

User InfoCatalog
User InfoCatalog - top level
Key Account Manager
Current Situation
420
Customer General Info
422
424
Internal Status
414
312
C-I
410

FIG. 4A

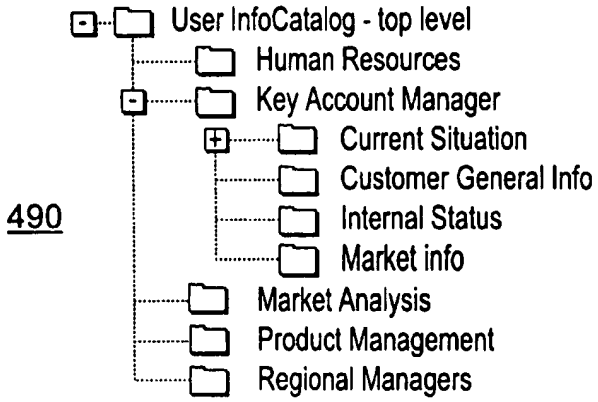

User InfoCatalog - top level
Human Resources
Key Account Manager
Current Situation
Customer General Info
Internal Status
Market info
Market Analysis
Product Management
Regional Managers
490

FIG. 4B

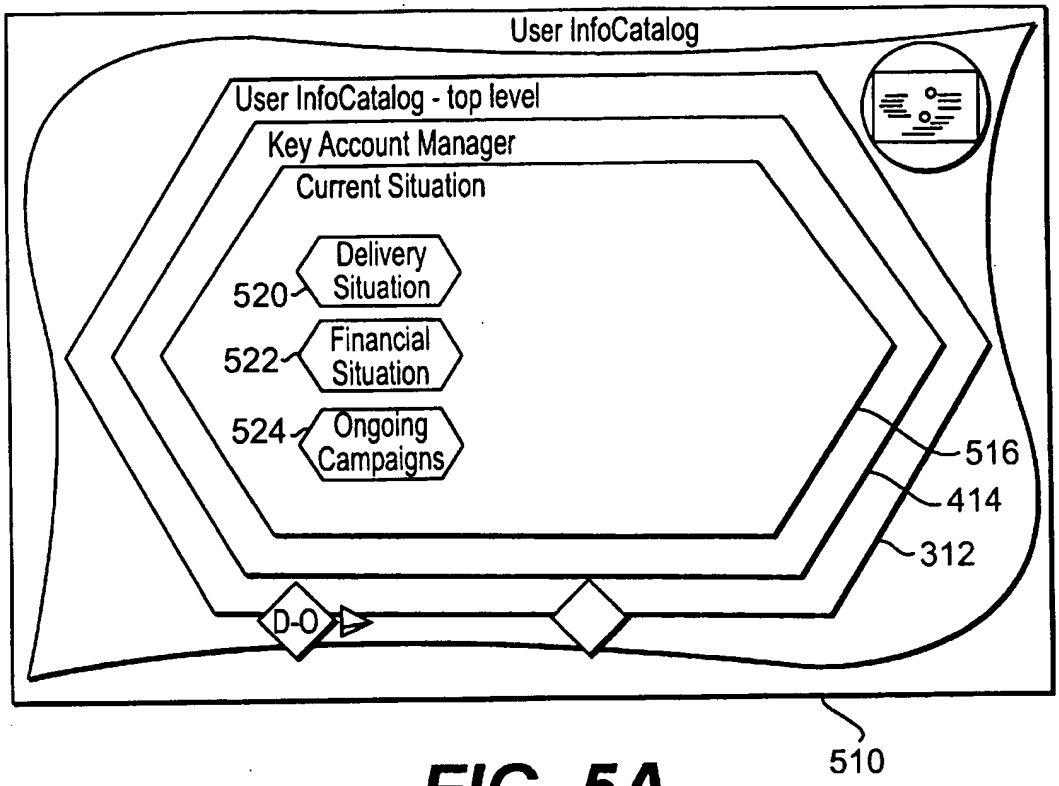

User InfoCatalog
User InfoCatalog - top level
Key Account Manager
Current Situation
Delivery Situation
520
Financial Situation
522
Ongoing Campaigns
524
516
414
312
D-O
510

FIG. 5A

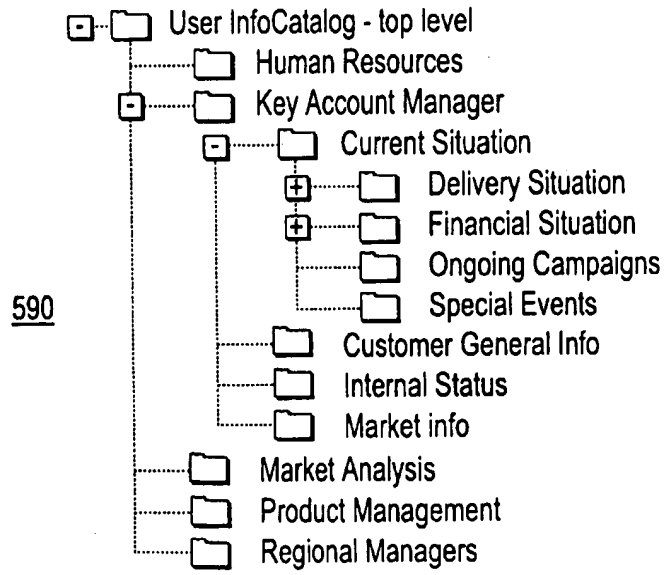

User InfoCatalog - top level
Human Resources
Key Account Manager
Current Situation
Delivery Situation
Financial Situation
Ongoing Campaigns
Special Events
Customer General Info
Internal Status
Market info
Market Analysis
Product Management
Regional Managers
590

FIG. 5B

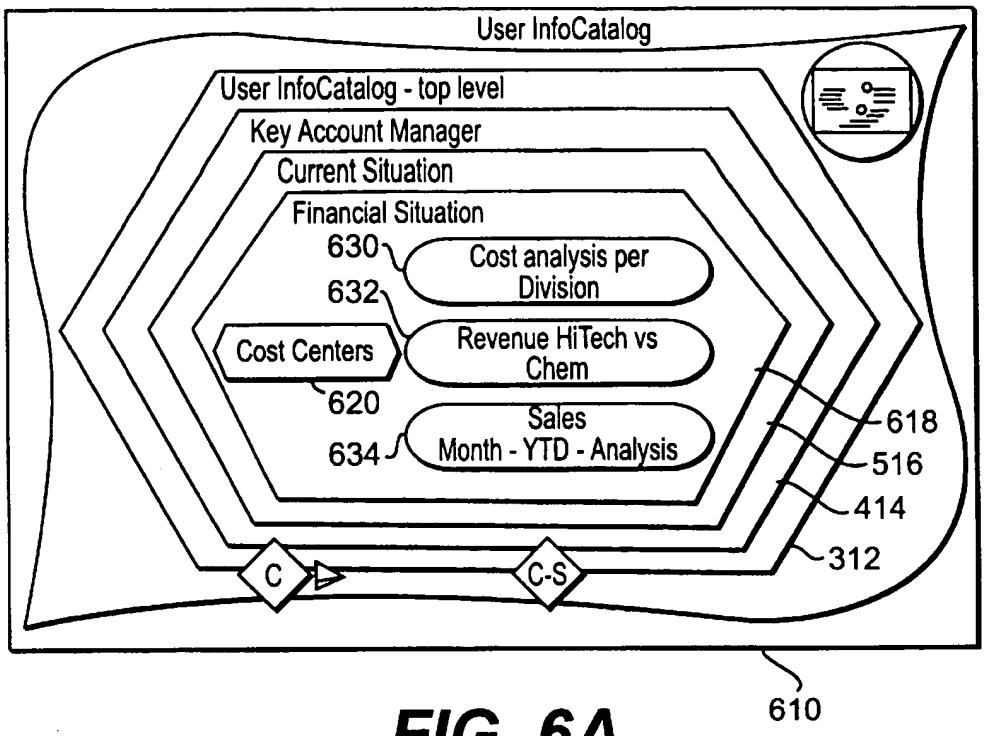

User InfoCatalog
User InfoCatalog - top level
Key Account Manager
Current Situation
Financial Situation
630
Cost analysis per Division
632
Cost Centers
Revenue HiTech vs Chem
620
634
Sales Month - YTD - Analysis
618
516
414
312
C
C-S
610

FIG. 6A

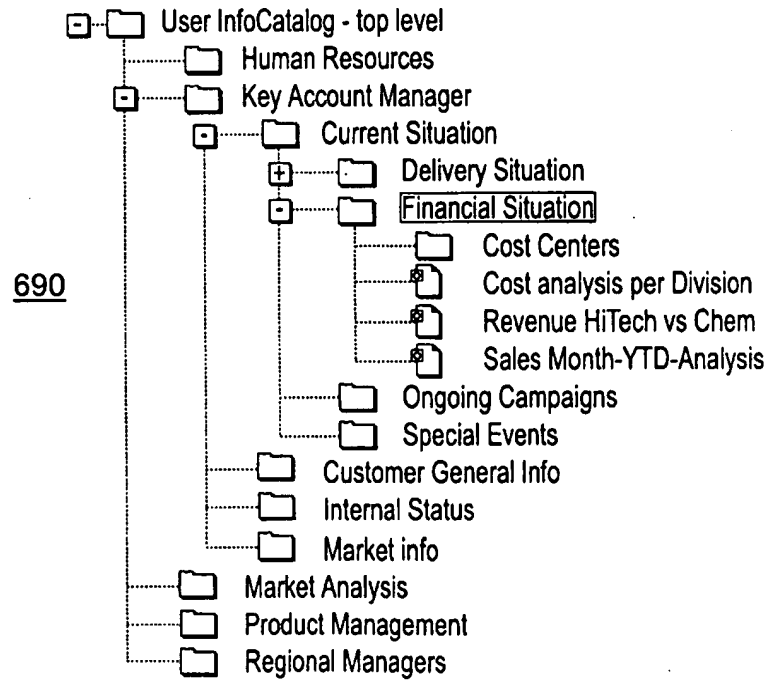

690
User InfoCatalog - top level
Human Resources
Key Account Manager
Current Situation
Delivery Situation
Financial Situation
Cost Centers
Cost analysis per Division
Revenue HiTech vs Chem
Sales Month-YTD-Analysis
Ongoing Campaigns
Special Events
Customer General Info
Internal Status
Market info
Market Analysis
Product Management
Regional Managers

FIG. 6B

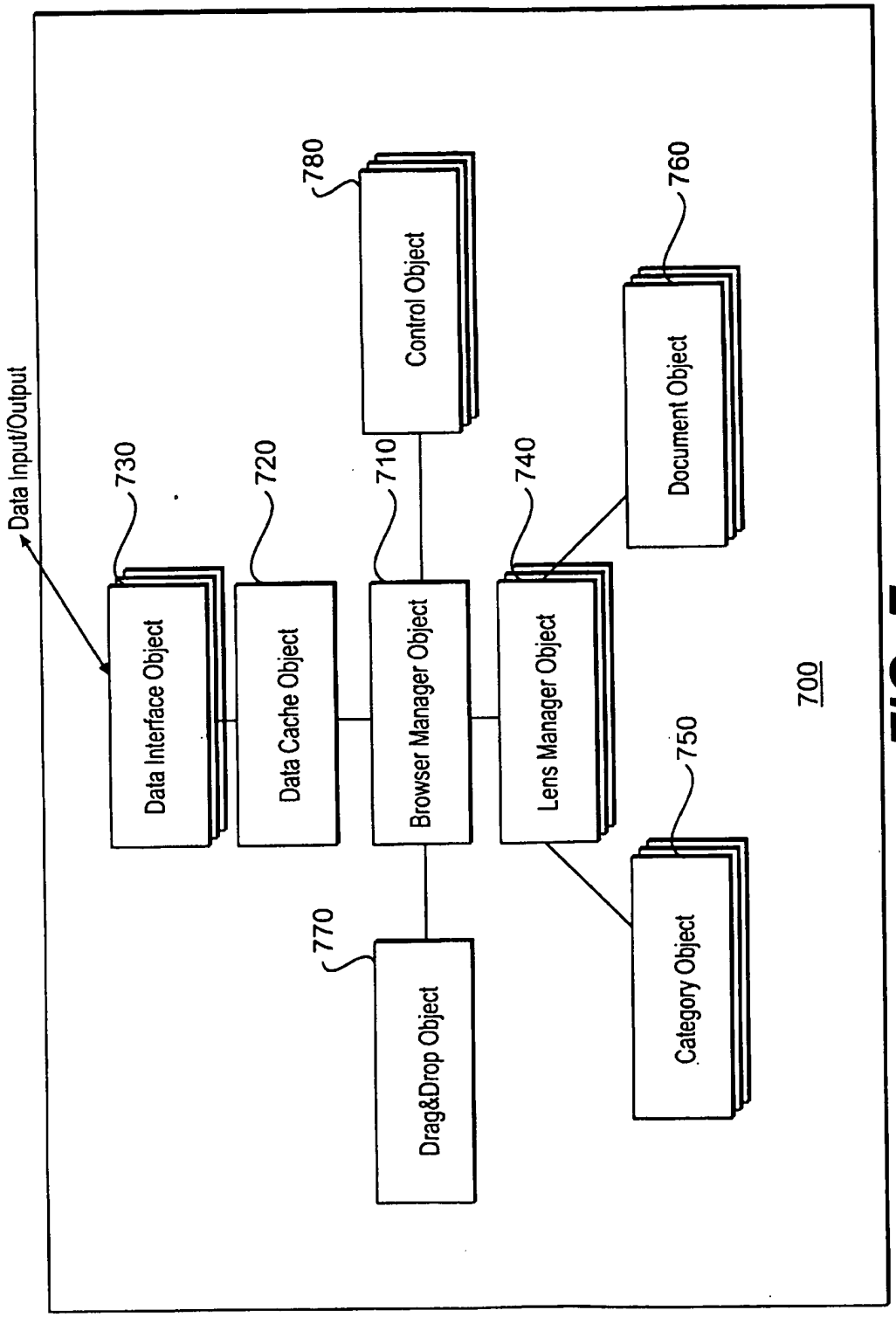

Data Input/Output
730
720
710
740
780
760
750
770
Data Interface Object
Data Cache Object
Browser Manager Object
Lens Manager Object
Control Object
Document Object
Category Object
Drag&Drop Object
700

FIG. 7

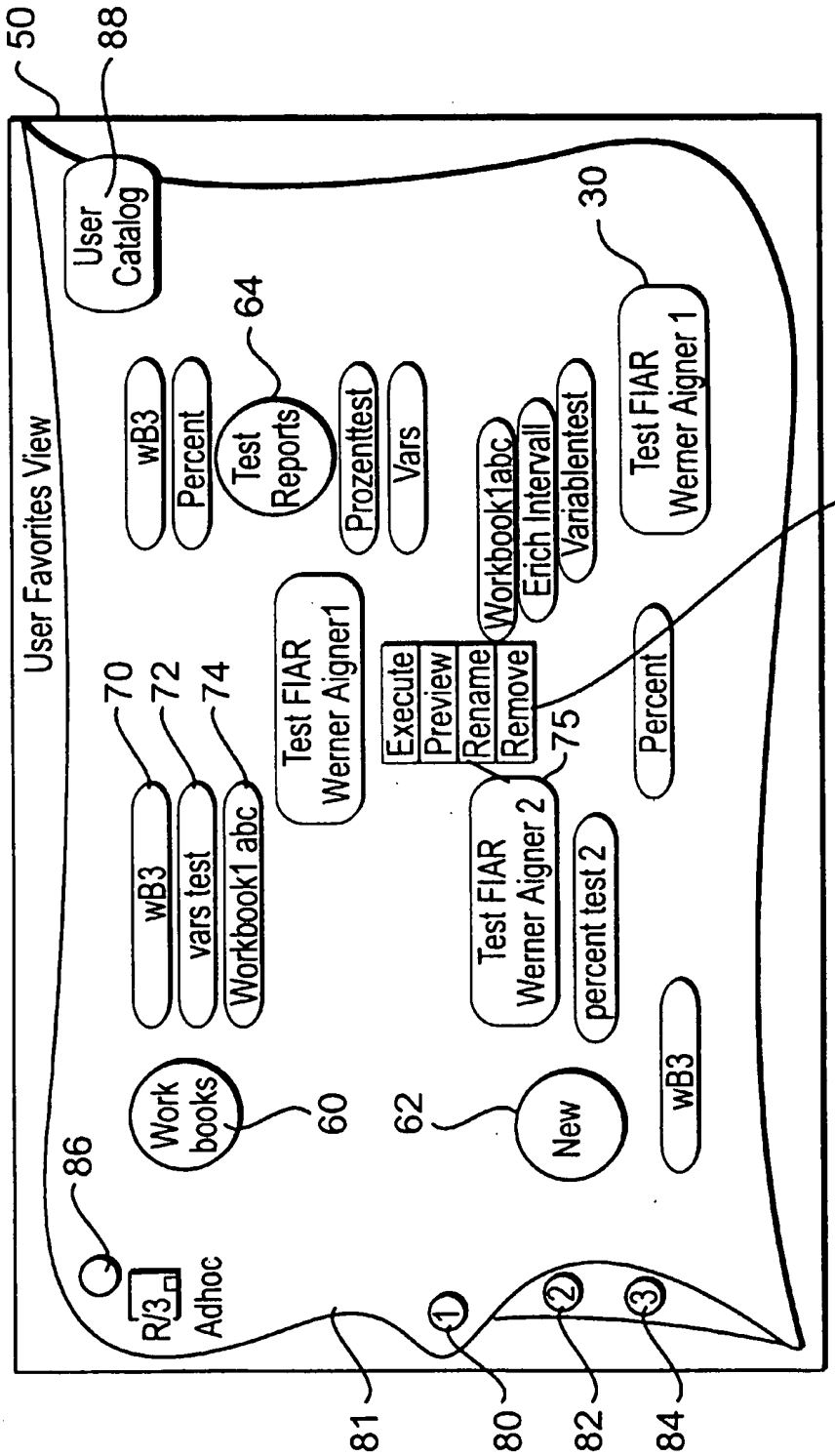

50
88
User Catalog
User Favorites View
wB3
Percent
64
Test Reports
Prozenttest
Vars
30
Test FIAR Werner Aigner 1
Workbook1abc
Erich Intervall
Variablentest
77
Test FIAR Werner Aigner1
Execute
Preview
Rename
Remove
75
70
72
74
wB3
vars test
Workbook1 abc
Work books
60
62
New
Test FIAR Werner Aigner 2
percent test 2
Percent
wB3
86
R/3
Adhoc
81
80
82
84

FIG. 8

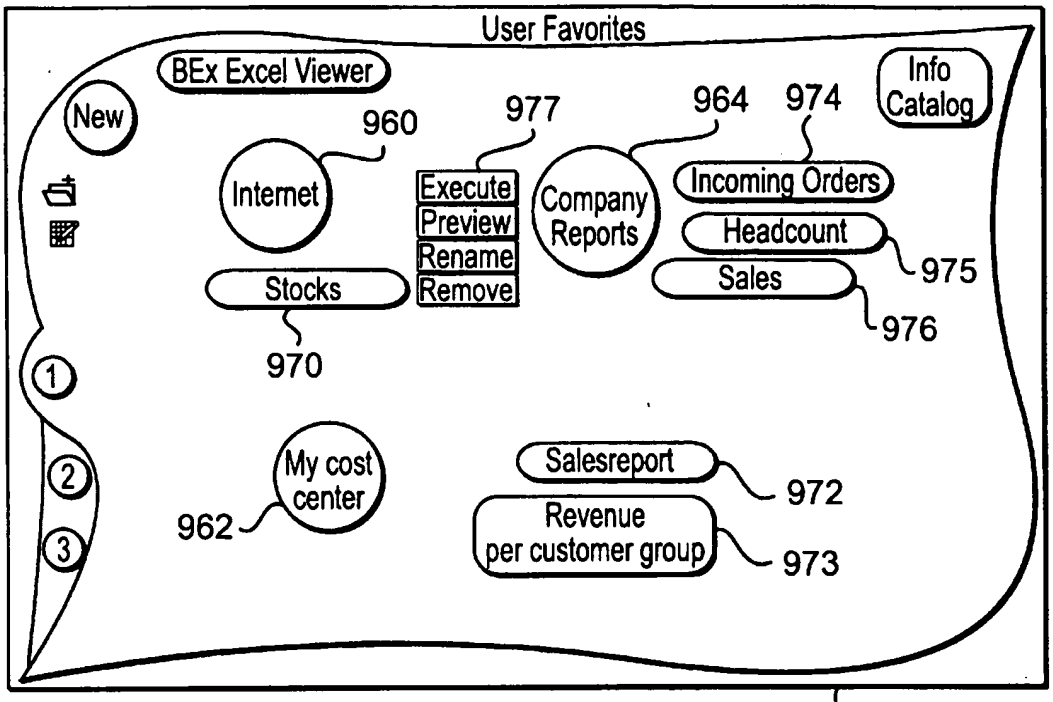

User Favorites
BEx Excel Viewer
New
960
977
964
974
Info Catalog
Internet
Execute
Preview
Rename
Remove
Company Reports
Incoming Orders
Headcount
975
Sales
976
Stocks
970
1
2
3
My cost center
962
Salesreport
972
Revenue per customer group
973
950

FIG. 9A

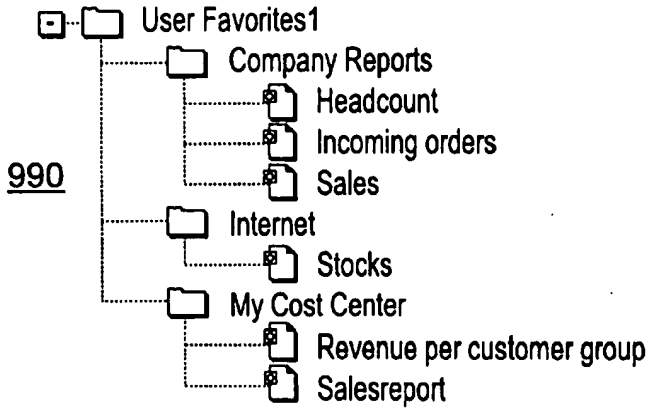

User Favorites1
Company Reports
Headcount
Incoming orders
Sales
990
Internet
Stocks
My Cost Center
Revenue per customer group
Salesreport

FIG. 9B

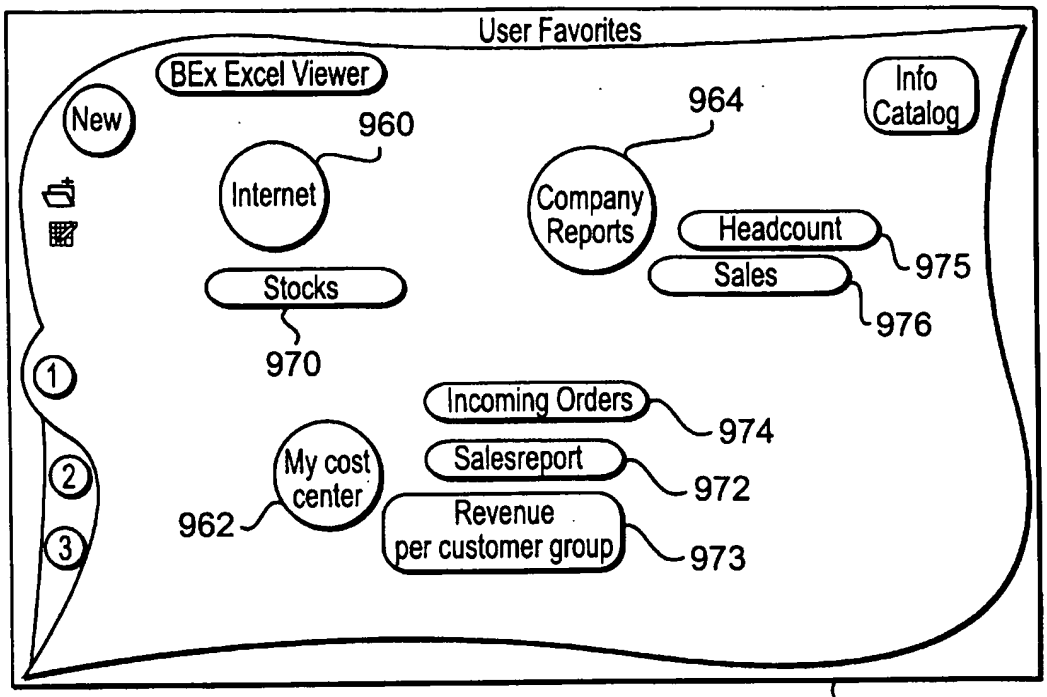


FIG. 10A

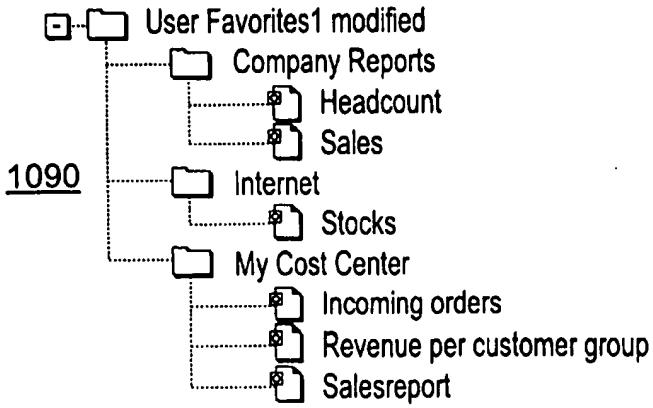


FIG. 10B

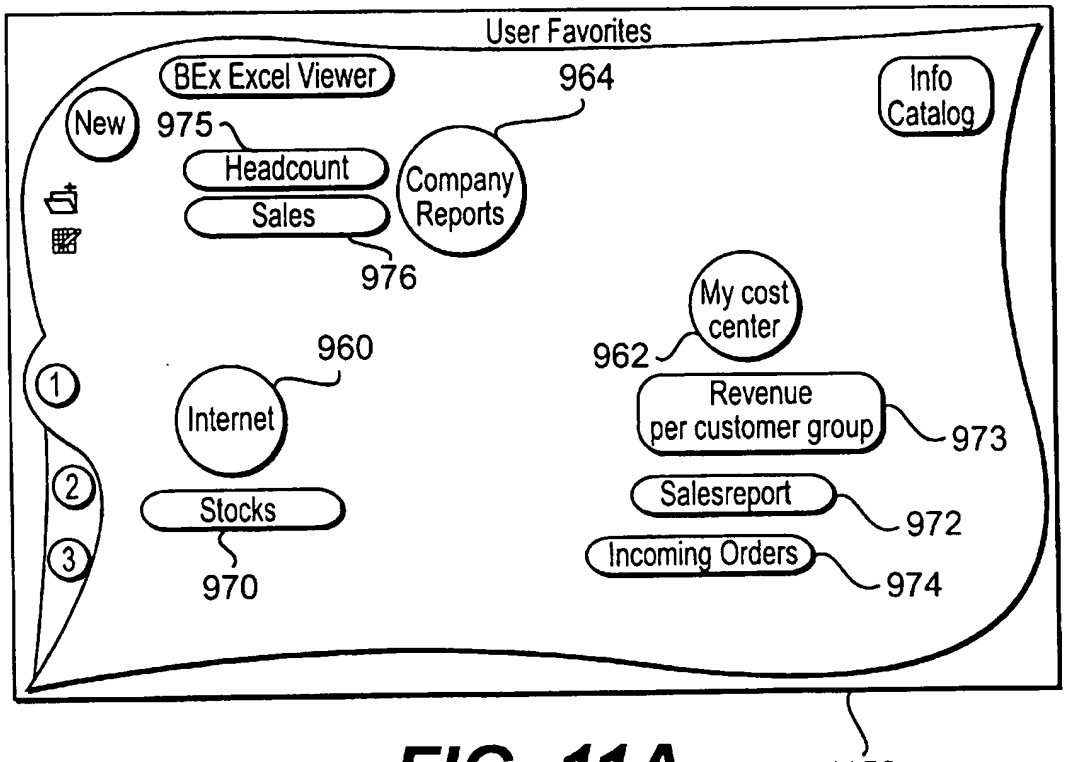


FIG. 11A

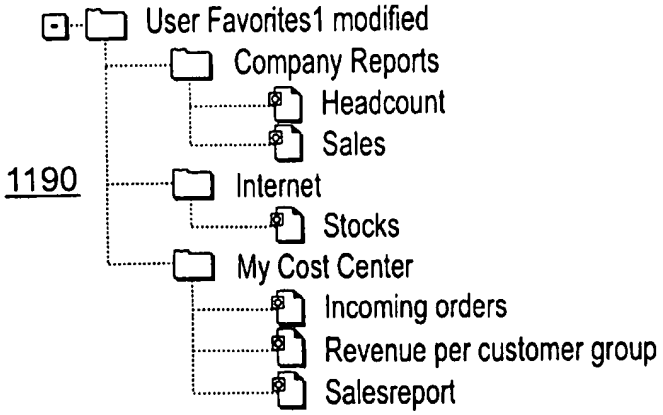


FIG. 11B

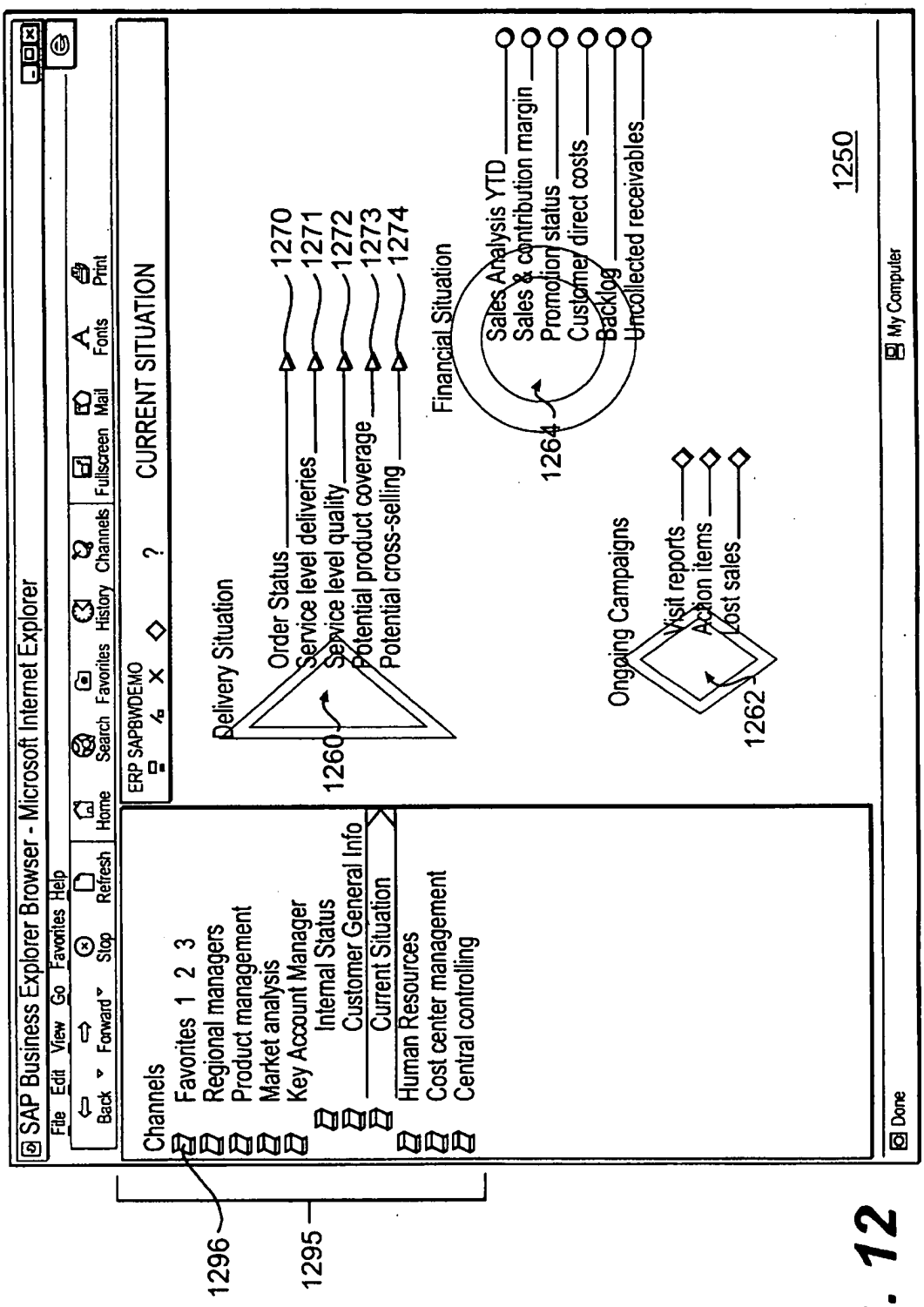

SAP Business Explorer Browser - Microsoft Internet Explorer
File Edit View Go Favorites Help
Back Forward Stop Refresh Home Search Favorites History Channels Fullscreen Mail Fonts Print
ERP SAPBWDEMO
CURRENT SITUATION
Channels
Favorites 1 2 3
Regional managers
Product management
Market analysis
Key Account Manager
Internal Status
Customer General Info
Current Situation
Human Resources
Cost center management
Central controlling
Delivery Situation
Order Status
Service level deliveries
Service level quality
Potential product coverage
Potential cross-selling
Financial Situation
Sales Analysis YTD
Sales & contribution margin
Promotion status
Customer direct costs
Backlog
Uncollected receivables
Ongoing Campaigns
Visit reports
Action items
Lost sales
1250
1260
1262
1264
1270
1271
1272
1273
1274
1295
1296
Done
My Computer

FIG. 12

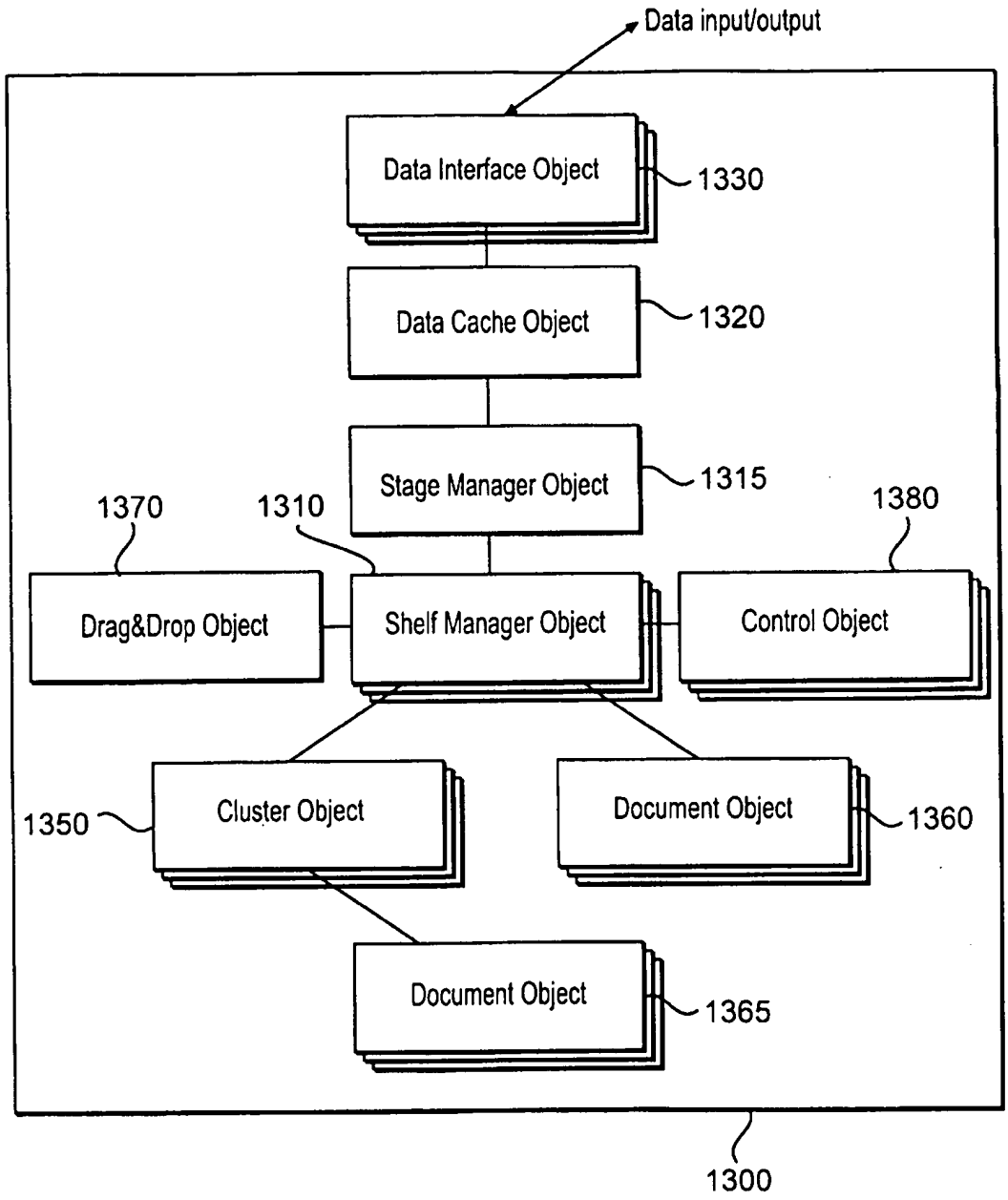

Data input/output
Data Interface Object
1330
Data Cache Object
1320
Stage Manager Object
1315
1370
1310
1380
Drag&Drop Object
Shelf Manager Object
Control Object
1350
Cluster Object
Document Object
1360
Document Object
1365
1300

FIG. 13

BROWSER FOR HIERARCHICAL STRUCTURES

This application claim benefit to provisional Application No. 60/058,946 Aug. 22, 1997.

BACKGROUND

With the tremendous advances in computer processing power and network capabilities over the past decade, the popularity of graphical user interfaces for viewing information over a network has increased substantially. Whereas in the past most networks used text displays to convey data from a server to a user, most interfaces today use graphics to convey information in a manner that is not only more pleasing to the eye, but also allows data to be manipulated and presented in ways not possible with pure text.

The first graphical user interfaces for networks were primitive and not entirely intuitive. A user would have to type complex commands to select the data to be displayed or to change the manner the data is displayed. Searches also required complex commands that were not very different from those used for text-based interfaces. However, searches were often the only way to find the desired data, since these primitive interfaces were not able to take full advantage of the links, to the extent that they existed, to quickly move a user from one set of data to another related set of data.

Today, browsers with graphical user interfaces are often used for viewing data across a network. Examples of popular browsers being used today are Netscape Navigator and Microsoft Explorer. Today's browsers can take greater advantage of links between related sets of data, and use these links to navigate among the various sets, including going up and down data hierarchies and revisiting previously viewed data.

While browsers and their graphical user interfaces are becoming increasingly sophisticated, the underlying data itself has become more complex. Data may be organized into huge databases of information that is spread across a network. Data may be organized into objects, including applications and related files, or groups thereof. This data may be organized in larger hierarchies, often many levels deep.

These hierarchically organized objects and groups usually consist of textual data, icons, or two-dimensional images. New development tools, such as JAVA, allow browsers to display more sophisticated objects, such as moving pictures or moving text. However, even these objects are limited by the underlying descriptive language used to create them. For example, today's browsers are not capable of displaying interactive VRML (Virtual Reality Markup Language) worlds.

It has become a challenge to present data in a clear and efficient manner through networks. Browsers can display only a limited amount of information on a user's screen at any given time. If a user chooses to use one lens, or window, to display a given set of data, the data disappears when a user subsequently displays other sets of data. If a user then wishes to revisit a prior set of data, it is usually necessary to visit all the links between the most recently viewed set of data and the desired prior set of data. Although most browsers allow users to create bookmarks for a given set of data, bookmarks are not automatically created, and require extra effort on a user's part. Many browsers also keep a list of recently accessed data, but users are then required to have the list displayed on the screen (taking up valuable screen space) and then remember which set of data is the desired set.

Some browsers allow users to have multiple lenses, or windows, each with a different set of data, displayed on the screen at any given time. However, as the multiple windows start overlapping each other, a user can easily lose track of the order in which the data was visited. Alternatively, a user may wish to recall the hierarchy of the sets of data previously visited, which is also easily forgotten when multiple windows are displayed.

On the other hand, a user may wish to have easy access to links to frequently accessed data. As noted above, most browsers allow a user to create bookmarks. However, as the number of bookmarks increases, it becomes more difficult for the user to quickly find a particular bookmark. This problem may be alleviated somewhat by having a hierarchical structure of bookmarks. However, this is far from a perfect solution, as an increasingly complicated hierarchical structure of bookmarks makes it even more difficult for a user to find a particular link, and time is wasted going through the various levels of the hierarchical structure. Thus, there is also a need to organize links in a manner that makes them readily accessible.

SUMMARY OF THE INVENTION

The present invention relates to an improvement over classic network browsers. The browser being disclosed provides easy access to previously accessed data.

It is an object of this invention to convey data in an efficient manner by displaying the data according to the data's hierarchical structure.

It is another object of this invention to give a user easy access to a hierarchical data structure by providing access to data at several levels within the hierarchical data structure simultaneously.

It is another object of this invention to provide labels for lenses displaying data at various levels within a hierarchical data structure.

It is another object of this invention to organize the display of hierarchical data in a logical manner that is pleasing to the eye.

It is another object of this invention to provide a means for a user to define a personal hierarchical data structure.

It is another object of the invention to use links in a personal hierarchical data structure in conjunction with natural association to access data in another hierarchical data structure having more levels than the personal hierarchical data structure.

It is another object of the invention to utilize colors, shapes, or other attributes to indicate natural association of links in a personal hierarchical data structure.

It is another object of the invention to use links in a personal hierarchical data structure in conjunction with natural association to access data, wherein the data is obtained through channels.

It is another object of the invention to allow exception reporting in conjunction with a personal hierarchical data structure.

It is another object of the invention to organize data into objects and groups of objects that can be represented as three-dimensional animations and VRML-worlds.

It is another object of the invention to display a context-sensitive menu when a cursor is placed on or near an object.

BRIEF DESCRIPTION OF THE DRAWINGS

FIG. 1 is a view of a computer screen showing an embodiment of the display for a browser.

FIG. 2 is a flow chart showing how the display of lenses in a browser is determined.

FIG. 3*a* is a view of a computer screen showing an example of a lens displaying a graphical representation of a category at a first level.

FIG. 3*b* is a tree structure representation of the category being represented in FIG. 3*a*.

FIG. 4*a* is a view of a computer screen showing an example of a lens displaying a graphical representation of a category at a second level.

FIG. 4*b* is a tree structure representation of the category being represented in FIG. 4*a*.

FIG. 5*a* is a view of a computer screen showing an example of a lens displaying a graphical representation of a category at a third level.

FIG. 5*b* is a tree structure representation of the category being represented in FIG. 5*a*.

FIG. 6*a* is a view of a computer screen showing an example of a lens displaying a graphical representation of a category at a fourth level.

FIG. 6*b* is a tree structure representation of the category being represented in FIG. 6*a*.

FIG. 7 is an example of a simplified object model showing how a hierarchical browser may be programmed.

FIG. 8 is a view of a computer screen showing an embodiment of the user favorites screen containing graphical representations of user-created links.

FIG. 9*a* is a view of a computer screen showing an embodiment of a display containing graphical representations of user-created links, wherein a popup menu is displayed.

FIG. 9*b* is a tree structure representation of the user-created links shown in FIG. 9*a*.

FIG. 10*a* is a view of a computer screen showing an embodiment of a display containing graphical representations of user-created links, wherein a link has been moved.

FIG. 10*b* is a tree structure representation of the user-created links shown in FIG. 10*a*.

FIG. 11*a* is a view of a computer screen showing an embodiment of a display containing graphical representations of user-created links, wherein the physical representations of the links have been moved on the display without changing the hierarchical structure of the links.

FIG. 11*b* is a tree structure representation of the user-created links shown in FIG. 11*a*.

FIG. 12 is a view of a computer screen showing an alternative embodiment of the user favorites screen containing graphical representations of user-created links.

FIG. 13 is an example of a simplified object model showing how a user favorites screen containing graphical representations of user-created links may be programmed.

DETAILED DESCRIPTION OF THE INVENTION

For clarity and simplicity, the invention will be described in the context of SAP software, including the marketed R/3 System. A brief description of the R/3 System follows.

R/3 can be described primarily as an online transaction processing system designed to provide integrated processing of all business routines and transactions. It includes enterprise-wide, integrated solutions, as well as specialized applications for individual, departmental functions. R/3 mirrors all of the business-critical processes of the enterprise—finance, manufacturing, sales, and human resources. It also offers various analytical capabilities to supplement the transaction processing function.

Recent paradigm shifts have forced companies to optimize all business processes along the net-value-added chain through the use of modern data processing and enterprise-wide information management. In order to become successful, companies often network with customers and suppliers such that products and services of high quality can be made available with minimal delay in response to customer demands. R/3 achieves these goals with online integration of data in the company.

The R/3 System is based on SAP's client/server architecture which separates the database, application, and presentation components for greater flexibility. This enables enterprises to take advantage of the various benefits of the architecture, including the capability to run across a variety of today's most popular UNIX-based hardware platforms. The R/3 System is designed so that the application systems and system functions are decoupled by a clear layer of architecture, with an application layer and a basis layer. The multilayer architecture of the R/3 System allows optimal load distribution, even in large installations with thousands of users. It has a modular structure with methods for controlling master-slave relationships between individual software components. Special servers linked by communications networks can be used for certain tasks without losing the integration of data and processes in the overall system. Implementation of separate servers for particular tasks makes optimal use of the performance potential and the different cost structures of available hardware architecture. This allows applications to be partitioned into different services and run on different servers. The fundamental services of the R/3 system (graphical presentation services, application services for handling of the application logic, database services for storage and recovery of business data, etc.) are partitioned and may evolve separately while maintaining interoperability.

It will be appreciated that the invention may be practiced using different hardware and different operating systems, and is not restricted to use with R/3. In fact, the invention is designed to be platform-independent.

Furthermore, the invention may be designed as modules that can be imbedded in a container. Thus, Active-X Control or OCX may be used, though the invention should not be construed as being limited to such implementations.

Preferred Embodiments of the Browser

It will be assumed that a hierarchically structured database exists. For illustrative purposes, within this database reside sets of data, including four sets of data with the following category names assigned to them: "User Info Catalog", "Companies", "Computer Technology", and "SAP". "User Info Catalog" is linked to "Companies", which is one hierarchical level below "User Info Catalog". Similarly, "Companies" is linked to "Computer Technology", which is another hierarchical level below. Finally, "Computer Technology" is linked to "SAP", which is a further hierarchical level below.

With reference to FIG. 1, screen **10** shows a preferred embodiment of the invention. Lenses **12**, **14**, **16**, and **18** display graphical representations of hierarchies of sets of data. Although the lenses illustrated in the figures are substantially hexagonal in shape, the lenses may be in the form of rectangles or other shapes in other preferred embodiments. Lens **12** is used to display a graphical repre-

sentation of the category "User Info Catalog". Since "Companies" is at a deeper hierarchical level than "User Info Catalog", the graphical representation of "Companies" is displayed in the smaller lens **14**. At the next level, lens **16** is used to display a graphical representation of "Computer Technology". Finally, lens **18** is used to display a graphical representation of "SAP". Each lens also displays the label or name of the category or set of data associated with that particular lens. Since lens **18** is the smallest lens displayed, it is also currently displaying a graphical representation of the set of data associated with "SAP". The set of data graphically represented in lens **18** contains objects, which may include text, icons, categories, applications, executable objects, such as executable reports, two-dimensional images, three-dimensional images, and VRML worlds.

Lens **18** is displaying a graphical representation of three objects **20**, **22**, and **24** that are categories one hierarchical level lower than "SAP". Objects **20**, **22**, and **24** are graphically represented as icons, but selecting one of these will cause a new, smaller lens to appear within lens **18** to display a graphical representation of the category selected. Lens **18** is also displaying graphical representations of objects **30**, **32**, **34**, and **36**. In a preferred embodiment, moving a cursor near one of these objects may cause a context-sensitive popup menu to appear with a list of possible commands to be selected. For example, moving a cursor near object **34** causes popup menu or context menu **37** to appear. Since object **34** is an executable object in the example, popup menu or context menu **37** has an "execute" option. In a preferred embodiment, a popup menu or context menu is predefined and is automatically displayed in a format based on the object type and user authorization. Optionally, such a menu could be hierarchical.

A user may go to a deeper level in the hierarchical data structure by selecting a set of data graphically represented on screen **10** that has a deeper hierarchical level than the set of data currently being graphically represented. For example, selecting one of objects **20**, **22**, or **24** will bring a user to a deeper level. In a preferred embodiment, a graphical representation of an object that is a category will indicate whether the category contains, at the next deeper level, an additional object. For example, symbol or icon **25** and symbol or icon **26** indicate that at least one object that is a category and at least one object that is not a category exist one hierarchical level deeper than object **20**. Similarly, symbol or icon **28** indicates that at least one category exists one hierarchical level deeper than object **22**.

There are numerous methods by which a user may select an object graphically represented on screen **10**. The most popular method used today is to use a mouse to move a mouse cursor to the graphical representation of the object being selected, and clicking on the mouse button. It will be appreciated that other methods, such as making use of a keyboard, are available.

Conversely, a user may wish to go to a higher level in the hierarchical data structure. This may be accomplished by selecting one of the lenses displayed behind the most recently displayed lens. For example, in FIG. **1** a user may select lens **12**, **14**, or **16** to display the contents of the selected lens. One way to select a lens that is already displayed on screen **10** is to move a mouse cursor to the displayed portion of the lens. For example, a user may view the objects previously displayed in lens **14** by placing the mouse cursor in the general area of point **15** and clicking the mouse button. Alternatively, a user may press a key or key combination on the keyboard, such as an <Alt> key and left arrow key combination, to go up one level. This works in a manner similar to the "Back" button found in many popular browsers today. Selecting a lens on screen **10** will displace all smaller lenses on screen **10**.

In a preferred embodiment, a user may navigate within a lens using a mouse, keyboard controls, or other means. For example, a mouse may be used to scroll in any direction within a lens. The left and right arrow keys of a keyboard may be used to scroll left or right, or to select objects having a graphical representation in a lens. Other keys may be used to select objects, for example by pressing the key corresponding to the first letter of the name of an object. In another preferred embodiment, buttons **40** and **42** may be used to scroll through objects. For example, selection of button **40** will cause the graphical representations of objects **20**, **22**, and **24** to disappear and to be replaced by graphical representations of other categories one hierarchical level lower than "SAP" to appear. Similarly, selection of button **42** will cause the graphical representations of objects **30**, **32**, **34**, and **36** to disappear and to be replaced by graphical representations of additional objects.

With reference to FIG. **2**, a preferred embodiment for determining the size of lenses is shown in flowchart **100**. A computer makes a request **102** for a set of data. Upon receipt **104** of the requested set of data, a comparison **106** is made between the hierarchical level of the received set of data and the hierarchical level of the set of data currently graphically represented within the smallest lens displayed on screen **10**. If the received set of data is at a deeper hierarchical level, the creation **108** of a lens that is smaller than the currently displayed lens or lenses is commenced. If the received set of data is not at a deeper hierarchical level, then removal **112** of the lens or lenses displaying graphical representations of any sets of data that are at the same hierarchical level as the received set of data or at a deeper hierarchical level than the received set of data is commenced. After removal **112** occurs, creation **108** of a lens that is smaller than the remaining displayed lens or lenses is commenced. After the creation **108** of a lens, display **110** of a graphical representation of the received set of data commences within the lens created in creation **108**. The computer is now ready to make a request **102** for another set of data.

It will be appreciated that flowchart **100** is only one preferred embodiment that there are other possible methods for determining the size of lenses and for displaying them in a fashion consistent with the invention. For example, in another preferred embodiment, if the computer has the information necessary to determine the hierarchical level of a requested set of data, it may be possible to make a comparison **106** before receipt **104** of the requested set of data. For example, if a requested set of data is associated with a URL that indicates the level of the data (such as a URL that uses slash characters to indicate a directory structure), it may be possible to determine the hierarchical level of the requested set of data before it arrives.

In another preferred embodiment, removal **112** of a displayed lens or lenses is not necessary if the creation **108** of a lens and display **110** of a graphical representation of the received set of data are accomplished in such a way that other lens or lenses are blocked off the screen **10**.

Examples of Browser Displaying Hierarchical Data

It will again be assumed that a hierarchically structured database exists. For illustrative purposes, within this database reside sets of data, including four sets of data with the following category names assigned to them: "User InfoCatalog", "Key Account Manager", "Current Situation",

and "Financial Situation". "User InfoCatalog" is linked to "Key Account Manager", which is one hierarchical level below "User InfoCatalog". Similarly, "Key Account Manager" is linked to "Current Situation", which is another hierarchical level below. Finally, "Current Situation" is linked to "Financial Situation", which is a further hierarchical level below.

With reference to FIG. **3***a*, screen **310** shows lens **312**, which is displaying a graphical representation of the category "User InfoCatalog". Lens **312** is displaying a graphical representation of three objects **320**, **322**, and **324** that are categories one hierarchical level lower than "User InfoCatalog". Objects **320**, **322**, and **324** are graphically represented as icons, but selecting one of these will cause a new, smaller lens to appear within lens **312** to display a graphical representation of the category selected. FIG. **3***b* shows a tree structure representation **390** of category or object "User InfoCatalog". Note that FIG. **3***b* indicates that additional objects or categories exist one hierarchical level lower than "User InfoCatalog", but their graphical representations are not currently shown on screen **310**. Selection of button **340**, which is analogous to button **40** described above, will cause the graphical representations of other categories one hierarchical level lower than "User InfoCatalog" to appear.

Suppose a user selects object **322**, which happens to be category "Key Account Manager". With reference to FIG. **4***a*, the result is screen **410** showing lens **414**, which is displaying a graphical representation of the category "Key Account Manager". Lens **414** is displaying a graphical representation of three objects **420**, **422**, and **424** that are categories one hierarchical level lower than "Key Account Manager". Objects **420**, **422**, and **424** are graphically represented as icons, but selecting one of these will cause a new, smaller lens to appear within lens **414** to display a graphical representation of the category selected. FIG. **4***b* shows a tree structure representation **490** of category or object "Key Account Manager".

Now suppose a user selects object **420**, which is category "Current Situation". With reference to FIG. **5***a*, the result is screen **510** showing lens **516**, which is displaying a graphical representation of the category "Current Situation". Lens **516** is displaying a graphical representation of three objects **520**, **522**, and **524** that are categories one hierarchical level lower than "Current Situation". Objects **520**, **522**, and **524** are graphically represented as icons, but selecting one of these will cause a new, smaller lens to appear within lens **516** to display a graphical representation of the category selected. FIG. **5***b* shows a tree structure representation **590** of category or object "Current Situation".

Finally, suppose a user selects object **522**, which is category "Financial Situation". With reference to FIG. **6***a*, the result is screen **610** showing lens **618**, which is displaying a graphical representation of the category "Financial Situation". Lens **618** is displaying a graphical representation of object **620** that is a category one hierarchical level lower than "Financial Situation". Lens **618** is also displaying graphical representations of objects **630**, **632**, and **634**, which, in a preferred embodiment, are not categories. FIG. **6***b* shows a tree structure representation **690** of category or object "Financial Situation".

While screen **610** is being displayed, a user may select another lens to return to a prior screen. By way of example, a user may select lens **414** (by, in a preferred embodiment, clicking in an area between the borders of lens **516** and lens **414**), resulting in screen **410** being displayed.

Example of a Simplified Object Model of the Browser

A method of creating the browser described above is to use object-oriented programming. A preferred method is to use an object-oriented programming editor, such as LINGO. It is to be appreciated by those skilled in the art that C++ and other languages having object-oriented programming capabilities may be used to create the browser described above. It should also be appreciated that programming languages and tools are constantly evolving, and such programming languages and tools may also be well-suited for the creation of the browser.

To avoid confusion, the word "Object" is capitalized in the following paragraph to distinguish it from the word "object" previously defined.

Referring to FIG. **7**, simplified Object Model **700** showing how the browser may be programmed is shown. Browser Manager Object **710** builds the graphic visualization of the browser and coordinates user interaction with browser controls. Data Cache Object **720** allows for temporary storage of sets of retrieved data, which are obtained through Data Interface Object **730**. Lens Manager Object **740** creates a lens for displaying graphical representations of sets of data. Such sets of data may include objects that are categories, which are controlled by Category Object **750**, and may include objects that are not categories, which objects are controlled by Document Object **760**. Drag and Drop Object **770** allows a user to select an object and drag it to another location (this is described below). Finally, Control Object **780** could coordinate the options or commands in a popup menu to be displayed, or any other user interaction, and coordinates the taking of the proper action when an option or command is selected.

It is to be appreciated that simplified Object Model **700** is but one way to organize the implementation of the browser.

A Preferred Embodiment of the User Favorites Screen

A user may wish to retain quick access to one or more objects without having to follow the links of the hierarchical data structure of the database. In a preferred embodiment, quick access is accomplished via a user favorites screen **50** shown in FIG. **8**. A preferred method for switching from screen **10** to user favorites screen **50** is by selecting cluster view icon **44** shown in FIG. **1**. A preferred method for switching from user favorites screen **50** to screen **10** is by selecting icon **88**. A preferred method alternative to these preferred methods is to display both screen **10** and user favorites screen **50** on one physical display simultaneously.

A link to an object may be created in user favorites screen **50** by the following preferred method. A user selects the desired object—object **32**, by way of example. The user places a mouse cursor on object **32**, presses on a mouse button, drags the cursor over to cluster view icon **44**, and releases the mouse button. User favorites screen **50** is then displayed, and the user moves the mouse cursor to a desired location on user favorites screen **50**. The user then releases the pressed mouse button. The process of selecting an object with a mouse cursor, pressing on a mouse button, dragging the cursor to another location, and releasing the mouse button is referred to as "drag and drop". It will be appreciated that there are numerous other methods by which a link to an object may be created in user favorites screen **50**.

In an alternative preferred embodiment, user favorites screen **50** may be used in conjunction with hierarchical structures without using the browser disclosed herein. For example, any hierarchical data controller that allows selection of objects within a hierarchical data structure may be used to select objects for the purpose of creating links to the selected objects in user favorites screen **50**. An alternative

preferred embodiment wherein a user favorites screen is used in conjunction with channels is described later.

User favorites screen **50** currently displays a graphical representation of a link to object **30**. This is an example of a graphical representation of a link to an object that currently also has a graphical representation on screen **10**. A user may access object **30** via screen **10** or via user favorites screen **50**. Optionally, the graphical representation of an object that also has a graphical representation of a link to the object on user favorites screen **50** receives a graphical attribute indicating that the object has been selected as a user favorite object.

In a preferred embodiment, a link created on user favorites screen **50** has the same graphical representation as the pertinent object on screen **10**. In an alternative preferred embodiment, the graphical representation of a link created on user favorite screen **50** inherits attributes, such as color, shading, or shape, associated with a cluster object link on user favorite screen **50**. By way of example, a user may use a mouse to drag the graphical representation of the link to object **30** over to the graphical representation of cluster object link **60** to cause the graphical representation of the link to object **30** to inherit the shading of the graphical representation of cluster object link **60**.

In a preferred embodiment, a user may create hierarchies for the links graphically represented on user favorites screen **50**. For example, a cluster object link, such as cluster object link **60**, **62**, or **64**, may be created by clicking on button **86**. The user may also type in a name for a cluster object link (such as the name "Workbooks" given to cluster object **60**). In the example on user favorites screen **50**, cluster object link **60** is one hierarchical level above object links **70, 72, 74** and is also linked to them. A link is created between a cluster object link, such as cluster object link **60**, and one or more object links, such as object links **70, 72, 74**, by using a mouse to drag the graphical representations of object links **70, 72,74** to the graphical representation of cluster object link **60**. Optionally, a user may create, move, rename, and remove cluster object links and object links. In a preferred embodiment, moving a cursor near the graphical representation of an object link may cause a context-sensitive popup menu to appear with a list of possible commands or options to be selected. For example, moving a cursor near the graphical representation of object link **75** causes popup menu or context menu **77** to appear. Since object link **75** is a link to an executable object in the example, popup menu or context menu **77** has an "execute" option and a "preview" option, in addition to "rename" and "remove" options. In a preferred embodiment, a popup menu or context menu is predefined and is automatically displayed in a format based on the object type and user authorization. For example, "remove" appears as an option only when there is authorization to remove the object link. Optionally, a popup menu or context menu could be hierarchical.

In a preferred embodiment, user favorites screen **50** may be saved (e.g., stored on a server and/or on computer-readable storage media) in such a way that the user-specified relative positions of the graphical representations of object links and cluster object links as they appear on user favorites screen **50** are also saved.

In a preferred embodiment, user favorites screen **50** may be used in conjunction with exception reporting. For example, an administrator may define an object to be associated with exception reporting, wherein certain conditions associated with an object will cause a user to be alerted to the occurrence of an exception. For example, a user may be alerted by having the graphical representation of an object link flash.

In a preferred embodiment, user favorites screen **50** is capable of displaying different shelves of graphical representations of links. In FIG. **8**, user favorites screen **50** is currently displaying a first shelf **81**. By selecting button **82** or button **84**, a user may select other shelves that display graphical representations of other links. By pressing button **80**, a user may then return to displaying shelf **81**. In a preferred embodiment, the graphical representation of an object link may be moved from one shelf to another shelf. A user may also move the graphical representation of a cluster object link from one shelf to another shelf, in which case the graphic representations of all the object links associated with that cluster object link also move to the other shelf. In another preferred embodiment, the shelf with the graphical representation of an object link associated with exception reporting may flash when alerting a user of an exception event.

Since the hierarchical organization of links whose graphical representations are displayed on user favorites screen **50** is displayed in a manner of natural association, the hierarchical nature of the links is not as readily perceived by a user. Since a user may create a cluster object link, such as cluster object link **60**, and cause object links, such as object links **70, 72,** and **74** to be linked to the cluster object link, a hierarchical relationship is created that is displayed as a collection of proximate graphical representations of cluster object links and object links on the screen. The examples below illustrate some of the advantages of natural association.

Examples of User Favorite Screens

The following examples illustrate the versatility of user favorites screens.

Referring to FIG. *9a*, user favorites screen **950** currently displays graphical representations of cluster object links **960, 962, 964**. Object link **970** is linked to cluster object link **960** (the display of popup menu or context menu **977** indicates that a cursor is near the graphical representation of object link **970**). Object links **972, 973** are linked to cluster object link **962**, while object links **974, 975, 976** are linked to cluster object link **964**. FIG. *9b* shows a tree structure representation **990** of the cluster object links and object links whose graphical representations are displayed on user favorites screen **950**. The graphical representations make the natural associations between the various groups of links apparent in a way that is not possible with a tree structure. The physical layout of the links, which may be defined by a user, obviates the need to display the hierarchical structure of the links.

In a preferred embodiment, if a user were to move the graphical representation of object link **974** by, for example, moving a cursor to the graphical representation of object link **974**, pressing a mouse button, dragging the graphical representation of object link **974** over to touch the graphical representation of cluster object link **962**, then dragging the graphical representation of object link **974** to a point on user favorites screen **950** that is just above the graphical representation of object link **972**, and releasing the mouse button, the result would look like user favorites screen **1050** in FIG. *10a*. This operation would remove the link between object link **974** and cluster object link **964**, and create a link between object link **974** and cluster object link **962**. FIG. *10b* shows a tree structure representation **1090** of the cluster object links and object links whose graphical representations are displayed on user favorites screen **1050**.

In a preferred embodiment, an attribute of the graphical representation of object link **974** would be altered by the

operation described above. By way of example only, assume that the graphical representation of cluster object link **962** is shaded red while the graphical representation of cluster object link **964** is shaded green. The graphical representation of object link **974** would be shaded green in user favorites screen **950**, but would automatically switch to a shade of red in user favorites screen **1050**.

To further illustrate the versatility of the user favorites screen, user favorites screen **1150** in FIG. **11***a* shows a different physical layout for the graphical representations of the links that are graphically represented in user favorites screen **1050**. FIG. **11***b* shows a tree structure representation **1190** of the cluster object links and object links whose graphical representations are displayed on user favorites screen **1150**. It is apparent that tree structure representation **1090** and tree structure representation **1190** are the same. When a user drags and drops the graphical representation of a cluster object link, the graphical representations of any object links that are linked to that cluster object link are also moved. For example, moving the graphical representation of cluster object link **960** from the top half of user favorites screen **1050** to the bottom half of user favorite screen **1150** causes the graphical representation of object link **970** to move as well. Furthermore, the drag and drop technique may also be used to move a graphical representation of an object link within the vicinity of a cluster object link. Thus, it is a simple matter to move the graphical representations of object links **975, 976** from the right of the graphical representation of cluster object link **964** (as in user favorites screen **1050**) to the left of the graphical representation of cluster object link **964** (as in user favorites screen **1150**). In a preferred embodiment, this operation would require two drag and drop operations. For example, a user may drag and drop the graphical representation of object link **975** and subsequently drag and drop the graphical representation of object link **976**. A further advantage of being able to customize the layout of the physical representation of objects is that a user, if it is so desired, may create a pseudo-hierarchy of object links simply by indenting graphical representations of certain object links relative to graphical representations of other object links.

Another Preferred Embodiment of the User Favorites Screen

An alternative embodiment of the user favorites screen is illustrated by user favorites screen **1250** in FIG. **12**. In this preferred embodiment, cluster object link **1260** is linked to object links **1270, 1271, 1272, 1273, 1274**. The graphical representations of these links share the same shape, which in user favorites screen **1250** is a triangular shape. The graphical representations of other cluster object links, and the graphical representations of object links linked to them, have other shapes. For example, the graphical representation of cluster object link **1262** has a diamond shape, and the graphical representation of cluster object link **1264** has a circular shape.

In a preferred embodiment, user favorites screen **1250** is used in conjunction with channels **1295**, which supply sets of data. Links to objects in public or semi-private channels may be created and organized utilizing a user favorites channel. A user may use icon **1296** to display a user favorites channel. For example, if a user clicks on icon **1296**, options to display user favorites channel **1**, user favorites channel **2**, or user favorites channel **3** appear (these options are analogous to buttons **80, 82, 84** used to select shelves for display in FIG. **8**).

A user may create a link to an object in another channel by dragging the object to icon **1296**. As with user favorites screen **50**, once an object link has been created and is graphically represented in user favorites screen **1250**, it may be linked to a cluster object link. It is to be appreciated that the operations that may be performed on user favorites screen **50** as described above may also be performed on user favorites screen **1250**.

Example of a Simplified Object Model of the User Favorites Screen

A method of creating the user favorites screen described above is to use object-oriented programming. A preferred method is to use an object-oriented programming editor, such as LINGO. It is to be appreciated by those skilled in the art that C++ and other languages having object-oriented programming capabilities may be used to create the cluster favorites screen described above. It should also be appreciated that programming languages and tools are constantly evolving, and such programming languages and tools may also be well-suited for the creation of the cluster favorites screen.

To avoid confusion, the word "Object" is capitalized in the following paragraph to distinguish it from the word "object" previously defined.

Referring to FIG. **13**, simplified Object Model **1300** showing how the user favorites screen may be programmed is shown. Shelf Manager Object **1310** controls all instances of cluster objects and objects associated with the currently displayed shelf or user favorites channel. Stage manager object **1315** controls all the visible action on the screen and coordinates event and information streams between the acting controls. Data Cache Object **1320** allows for temporary storage of sets of retrieved data, which are obtained through Data Interface Object **1330**. Cluster Object **1350** is an instance of a cluster object link, while Document Object **1360** is an instance of an object link that is not linked to a cluster object link. Document Object **1365** is an instance of an object link that is linked to a cluster object link. Drag and Drop Object **1370** allows a user to select an object or cluster object and drag it to another location. Finally, Control Object **1380** coordinates the options or commands in a popup menu to be displayed, or any other user interaction, and coordinates the taking of the proper action when an option or command is selected.

It is to be appreciated that simplified Object Model **1300** is but one way to organize the implementation of the user favorites screen.

Alternative Preferred Embodiments

In a preferred embodiment, the sets of data described above reside across a network of computers. The sets of data that reside on the Internet, or any sets of data that reside in more than one database, may be considered parts of one large database. In an alternative preferred embodiment, sets of data may reside on one computer. It will be appreciated that the invention may be practiced on one computer or over a network of computers. It will also be appreciated that a computer may receive sets of data from its own storage medium or cache, and is not limited to receiving data from other computers.

While there have been shown and described and pointed out fundamental novel features of the invention as applied to embodiments thereof, it will be understood that various omissions and substitutions and changes in the form and details of the invention, as herein disclosed, may be made by those skilled in the art without departing from the spirit of the invention. It is expressly intended that all combinations

of those elements and/or method steps which perform substantially the same function in substantially the same way to achieve the same results are within the scope of the invention. It is the intention, therefore, to be limited only as indicated by the scope of the claims appended hereto.

What is claimed is:

1. A method of displaying and interacting with links to data, which data is organized in a hierarchical data structure, the method comprising the steps of:

(a) sending a first request from a graphical user interface on a computer to provide a first set of data, said first set of data being in said hierarchical data structure and having a position in said hierarchical data structure and having a associated with said position of said first set of data in said hierarchical data structure;

(b) receiving in said computer said first set of data in response to said first request;

(c) converting said first set of data to a graphical representation for visual display;

(d) displaying said graphical representation for visual display of said first set of data in a first lens displayed by said graphical user interface;

(e) sending a second request from said graphical user interface to provide a second set of data, said second set of data being in said hierarchical data structure and having a position in said hierarchical data structure and having a level associated with said position of second set of data in said hierarchical data structure;

(f) receiving in said computer said second set of data in response to said second request;

(g) comparing said level of said second set of data with said level of said first set of data to determine which level is deeper within said hierarchical data structure;

(h) converting said second set of data to a graphical representation for visual display based upon said comparing of said level of said second set of data with said level of said first set of data; and

(i) displaying said graphical representation for visual display of said second set of data in a second lens displayed by said graphical user interface stacked within and on top of said first lens responsive only to a determination that said level of said second set of data is deeper than said level of said first set of data in said hierarchical data structure.

2. The method of displaying and interacting with links to data of claim 1, wherein said first lens and said second lens are substantially hexagonal in shape.

3. The method of displaying and interacting with links to data of claim 1, wherein at least said second set of data contains a representation of a three-dimensional animation.

4. The method of displaying and interacting with links to data of claim 1, wherein at least said second set of data contains a representation of a virtual reality world.

5. The method of claim 1, wherein the sending of said requests and said receiving of said sets of data is via a computer network.

6. The method of claim 5, wherein the computer network is the Internet.

7. The method of claim 1, and further comprising:

discontinuing display of said graphical representation for visual display of said first set of data by said graphical user interface responsive to a determination that said level of said second set of data is equal to said level of said first set of data in said hierarchical data structure.

8. A method of displaying and interacting with links to data, which data is organized in a hierarchical data structure, the method comprising the steps of:

(a) sending a first request from a graphical user interface on a computer to provide a first set of data, said first set of data being in said hierarchical data structure and having a position in said hierarchical data structure and having a level associated with said position of said first set of data in said hierarchical data structure;

(b) receiving in said computer said first set of data in response to said first request;

(c) converting said first set of data to a graphical representation for visual display;

(d) displaying said graphical representation for visual display of said first set of data in a first lens displayed by said graphical user interface;

(e) sending a second request from said graphical user interface to provide a second set of data, said second set of data being in said hierarchical data structure and having a position in said hierarchical data structure and having a level associated with said position of said second set of data in said hierarchical data structure;

(f) receiving in said computer said second set of data in response to said second request;

(g) comparing said level of said second set of data with said level of said first set of data to determine which level is deeper within said hierarchical data structure;

(h) converting said second set of data to a graphical representation for visual display based upon said comparing said level of said second set of data with said level of said first set of data; and

(i) displaying said graphical representation for visual display of said second set of data in a second lens displayed by said graphical user interface stacked within and on top of said first lens only when said level of said second set of data is deeper than said level of said first set of data in said hierarchical data structure; and wherein:

at least one of said graphical representation for visual display of said first set of data and said graphical representation for visual display of said second set of data contains a graphical representation for visual display of a category;

said category has a position in said hierarchical data structure and has a level associated with said position of said category; and

said graphical representation for visual display of said category indicates whether a third set of data having a position at a deeper level than said level associated with said position of said category exists.

9. The method of claim 8, wherein the sending of said requests and said receiving of said sets of data is via a computer network.

10. The method of claim 9, wherein the computer network is the Internet.

11. The method of claim 8, and further comprising:

discontinuing display of said graphical representation for visual display of said first set of data by said graphical user interface responsive to a determination that said level of said second set data is equal to said level of said first set of data in said hierarchical data structure.

12. A method of displaying and interacting with links to data in an R/3 system, which data is organized in a hierarchical data structure, the method comprising the steps of:

(a) sending a first request from an R/3 graphical user interface on a computer to provide a first set of data, said first set of data being in said hierarchical data structure and having a position in said hierarchical data

structure and having a level associated with said position of first set of data in said hierarchical data structure;

(b) receiving in said computer said first set of data in response to said first request;

(c) converting said first set of data to a graphical representation for visual display;

(d) displaying said graphical representation for visual display of said first set of data in a first lens displayed by said R/3 graphical user interface;

(e) sending a second request from said R/3 graphical user interface to provide a second set of data, said second set of data being in said hierarchical data structure and having a position in said hierarchical data structure and having a level associated with said position of second set of data in said hierarchical data structure;

(f) receiving in said computer said second set of data in response to said second request;

(g) comparing said level of said second set of data with said level of said first set of data to determine which level is deeper within said hierarchical data structure;

(h) converting said second set of data to a graphical representation for visual display based upon said comparing of said level of said second set of data with said level of said first set of data; and

(i) displaying said graphical representation for visual display of said second set of data in a second lens displayed by said R/3 graphical user interface stacked within and on top of said first lens responsive only to a determination from said comparing that said level of said second set of data is deeper than said level of said first set of data in said hierarchical data structure.

13. The method of claim 12, wherein the sending of said requests and said receiving of said sets of data is via a computer network.

14. The method of claim 13, wherein the computer network is the Internet.

15. The method of claim 12, and further comprising:

discontinuing display of said graphical representation for visual display of said first set of data by said graphical user interface responsive to a determination that said level of said second set of data is equal to said level of said first set of data in said hierarchical data structure.

16. A computer system comprising a display, a central processor, a graphics processor, a memory, an input device, and a graphical user interface, wherein:

said memory has a hierarchical data structure, said hierarchical data structure having a first set of data and a second set of data, said first set of data having a position in said hierarchical data structure and having a level associated with said position of said first set of data in said hierarchical data structure, said second set of data having a position in said hierarchical data structure and having a level associated with said position of said second set of data in said hierarchical data structure;

said graphical user interface requests said first set of data;

said graphics processor receives said first set of data in response to said request for said first set of data;

said graphics processor converts said first set of data to a graphical representation for visual display;

said graphical user interface displays said graphical representation for visual display of said first set of data in a first lens on said display;

said graphical user interface requests said second set of data;

said graphical processor receives said second set of data in response to said request for said second set of data;

said central processor compares said level of said second set of data with said level of said first set of data to determine which level is deeper within said hierarchical data structure;

said graphics processor converts said second set of data to a graphical representation for visual display based upon said comparing of said level of said second set of data with said level of said first set of data; and

said graphical user interface displays said graphical representation for visual display of said second set of data in a second lens stacked within and on top of said first lens on said display responsive only to a determination that said level of said second set of data is deeper than said level of said first set of data.

17. The computer system of claim 16, wherein said first lens and said second lens are substantially hexagonal in shape.

18. The computer system of claim 16, wherein said central processor and said graphics processor are the same processor.

19. The computer system of claim 16, wherein at least said second set of data contains a representation of a three-dimensional animation.

20. The computer system of claim 16, wherein at least said second set of data contains a representation of a virtual reality world.

21. The computer system of claim 16, wherein said graphical user interface operates in conjunction with an instance of SAP Business Information Warehouse.

22. The computer system of claim 16, wherein said graphical user interface displays a context-sensitive pop-up menu, based on attributes of said second set of data and based on an authorization level.

23. The computer system of claim 16, wherein said graphical user interface is imbedded in a container.

24. The system of claim 16, wherein the sets of data are requested and received over a computer network.

25. The system of claim 24, wherein the computer network is the Internet.

26. The system of claim 16, wherein said graphical user interface discontinues display of said graphical representation for visual display of said first set of data responsive to a determination that said level of said second set of data is equal to said level of said first set of data in said hierarchical data structure.

27. A computer system comprising a display, a central processor, a graphics processor, a memory, an input device, and a graphical user interface, wherein:

said memory has a hierarchical data structure, said hierarchical data structure having a first set of data and a second set of data, said first set of data having a position in said hierarchical data structure and having a level associated with said position of said first set of data in said hierarchical data structure, said second set of data having a position in said hierarchical data structure and having a level associated with said position of said second set of data in said hierarchical data structure;

said graphical user interface requests said first set of data;

said graphics processor receives said first set of data in response to said request for said first set of data;

said graphics processor converts said first set of data to a graphical representation for visual display;

said graphical user interface displays said graphical representation for visual display of said first set of data in a first lens on said display;

said graphical user interface requests said second set of data;

said graphical processor receives said second set of data in response to said request for said second set of data;

said central processor compares said level of said second set of data with said level of said first set of data to determine which level is deeper within said hierarchical data structure;

said graphics processor converts said second set of data to a graphical representation for visual display based upon said comparing said level of said second set of data with said level of said first set of data; and

said graphical user interface displays said graphical representation for visual display of said second set of data in a second lens stacked within and on top of said first lens on said display only when said level of said second set of data is deeper than said level of said first set of data; and wherein:

at least one of said graphical representation for visual display of said first set of data and said graphical representation for visual display of said second set of data contains a graphical representation for visual display of a category;

said category has a position in said hierarchical data structure and has a level associated with said position of said category; and

said graphical representation for visual display of said category indicates whether a third set of data having a position at a deeper level than said level associated with said position of said category exists.

28. The system of claim **27**, wherein the sets of data are requested and received over a computer network.

29. The system of claim **28**, wherein the computer network is the Internet.

30. The system of claim **27**, wherein said graphical user interface discontinues display of said graphical representation for visual display of said first set of data responsive to a determination that said level of said second set of data is equal to said level of said first set of data in said hierarchical data structure.

31. A method of displaying sets of data organized in a hierarchical data structure, the method comprising displaying a plurality of lenses, each lens having a size, wherein:

each said lens is associated with a respective set of data, each said set of data being in said hierarchical data structure and each having a position in said hierarchical data structure and each having a level associated with said position of said set of data in said hierarchical data structure; and

said size of each said lens being determined based on said level of said set of data associated with said lens so that one of said lenses is smaller than another of said lenses where said one of said lenses is associated with a set of data having a deeper level in the hierarchical data structure than the level of the set of data associated with said another of said lenses.

32. The method of displaying data organized in a hierarchical data structure of claim **31**, wherein each said lens has a label.

33. The method of displaying data organized in a hierarchical data structure of claim **31**, further comprising selecting any one of said plurality of lenses using a graphical user interface, displaying said any one of said plurality of lenses, and displaying said set of data associated with said any one of said plurality of lenses.

34. The method of displaying data organized in a hierarchical data structure of claim **31**, wherein at least said one of said lenses is associated with data containing a representation of a three-dimensional animation.

35. The method of displaying data organized in a hierarchical data structure of claim **31**, wherein at least said one of said lenses is associated with data that contains a representation of a virtual reality world.

36. The method of displaying data organized in a hierarchical data structure of claim **31**, wherein each said lens is substantially hexagonal in shape.

37. The method of claim **31**, wherein the sets of data are requested and received from the hierarchical data structure over a computer network.

38. The method of claim **37**, wherein the computer network is the Internet.

39. The method of claim **31**, and further comprising:

discontinuing display of one of said lenses responsive to a determination that said level of the set of data associated therewith is equal to said level of the set of data associated with another lens to be displayed.

40. A computer system for displaying and interacting with data organized in a hierarchical data structure, which comprises:

(a) means for sending a first request from a graphical user interface on a computer to provide a first set of data, said first set of data being in said hierarchical data structure and having a position in said hierarchical data structure and having a level associated with said position of said first set of data in said hierarchical data structure;

(b) means for receiving in said computer said first set of data in response to said first request;

(c) means for converting said first set of data to a graphical representation for visual display;

(d) means for displaying said graphical representation for visual display of said first set of data in a first lens displayed by said graphical user interface;

(e) means for sending a second request from said graphical user interface to provide a second set of data, said second set of data being in said hierarchical data structure and having a position in said hierarchical data structure and having a level associated with said position of second set of data in said hierarchical data structure;

(f) means for receiving in said computer said second set of data in response to said second request;

(g) means for comparing said level of said second set of data with said level of said first set of data to determine which level is deeper within said hierarchical data structure;

(h) means for converting said second set of data to a graphical representation for visual display based upon said comparing of said level of said second set of data with said level of said first set of data; and

(i) means for displaying said graphical representation for visual display of said second set of data in a second lens displayed by said graphical user interface stacked within and on top of said first lens responsive only to a determination that said level of said second set of data is deeper than said level of said first set of data in said hierarchical data structure.

41. The computer system of claim **40**, wherein said first lens and said second lens are substantially hexagonal in shape.

42. The computer system of claim **40**, wherein at least said second set of data contains a representation of a three-dimensional animation.

43. The computer system of claim 40, wherein at least said second set of data contains a representation of a virtual reality world.

44. The computer system of claim 40, wherein said graphical user interface operates in conjunction with an instance of SAP Business Information Warehouse.

45. The computer system of claim 40, wherein said graphical user interface displays a context-sensitive pop-up menu, based on attributes of said second set of data and based on an authorization level.

46. The computer system of claim 40, wherein said graphical user interface is imbedded in a container.

47. The system of claim 40, wherein the sets of data are requested and received over a computer network.

48. The system of claim 47, wherein the computer network is the Internet.

49. The system of claim 40, and further comprising:

means for discontinuing display of said graphical representation for visual display of said first set of data by said graphical user interface responsive to a determination that said level of said second set of data is equal to said level of said first set of data in said hierarchical data structure.

50. A computer system for displaying and interacting with data organized in a hierarchical data structure, which comprises:

(a) means for sending a first request from a graphical user interface on a computer to provide a first set of data, said first set of data being in said hierarchical data structure and having a position in said hierarchical data structure and having a level associated with said position of said first set of data in said hierarchical data structure;

(b) means for receiving in said computer said first set of data in response to said first request;

(c) means for converting said first set of data to a graphical representation for visual display;

(d) means for displaying said graphical representation for visual display of said first set of data in a first lens displayed by said graphical user interface;

(e) means for sending a second request from said graphical user interface to provide a second set of data, said second set of data being in said hierarchical data structure and having a position in said hierarchical data structure and having a level associated with said position of said second set of data in said hierarchical data structure;

(f) means for receiving in said computer said second set of data in response to said second request;

(g) means for comparing said level of said second set of data with said level of said first set of data to determine which level is deeper within said hierarchical data structure;

(h) means for converting said second set of data to a graphical representation for visual display based upon said comparing said level of said second set of data with said level of said first set of data; and

(i) means for displaying said graphical representation for visual display of said second set of data in a second lens displayed by said graphical user interface stacked within and on top of said first lens only when said level of said second set of data is deeper than said level of said first set of data in said hierarchical data structure; and wherein:

at least one of said graphical representation for visual display of said first set of data and said graphical representation for visual display of said second set of data contains a graphical representation for visual display of a category;

said category has a position in said hierarchical data structure and has a level associated with said position of said category; and

said graphical representation for visual display of said category indicates whether a third set of data having a position at a deeper level than said level associated with said position of said category exists.

* * * * *

UNITED STATES PATENT AND TRADEMARK OFFICE

CERTIFICATE OF CORRECTION

PATENT NO. : 6,278,991 B1
DATED : August 21, 2001
INVENTOR(S) : Peter Ebert

It is certified that error appears in the above-identified patent and that said Letters Patent is hereby corrected as shown below:

Column 13,
Line 14, before "associated" insert -- level --.
Line 27, before "second" insert -- said --.

Column 15,
Line 2, before "first" insert -- said --.
Line 15, before "second" insert -- said --.

Column 16,
Line 1, change "graphical" to -- graphics --.

Column18,
Line 42, before "second" insert -- said --.

Signed and Sealed this

Eighteenth Day of June, 2002

Attest:

Attesting Officer

JAMES E. ROGAN
Director of the United States Patent and Trademark Office

US006496208B1

(12) **United States Patent**
Bernhardt et al.

(10) **Patent No.: US 6,496,208 B1**
(45) **Date of Patent: Dec. 17, 2002**

(54) **METHOD AND APPARATUS FOR VISUALIZING AND EXPLORING LARGE HIERARCHICAL STRUCTURES**

(75) Inventors: **Jeff Bernhardt**, Woodinville; **Usama M. Fayyad**, Mercer Island, both of WA (US)

(73) Assignee: **Microsoft Corporation**, Redmond, VA (US)

(*) Notice: Subject to any disclaimer, the term of this patent is extended or adjusted under 35 U.S.C. 154(b) by 0 days.

(21) Appl. No.: **09/300,932**

(22) Filed: **Apr. 28, 1999**

Related U.S. Application Data

(60) Provisional application No. 60/099,685, filed on Sep. 10, 1998.

(51) **Int. Cl.**[7] **G06F 3/14**

(52) **U.S. Cl.** **345/853**; 345/764; 345/804; 345/781; 345/805

(58) **Field of Search** 345/853, 835, 345/804, 841, 781, 968, 805, 767, 764, 733, 765

(56) **References Cited**

U.S. PATENT DOCUMENTS

5,295,244	A	*	3/1994	Dev et al.	345/357
5,309,555	A	*	5/1994	Akins et al.	395/330
5,528,735	A	*	6/1996	Strasnick et al.	395/127
5,571,965	A	*	11/1996	Rogers et al.	73/462
5,572,640	A	*	11/1996	Schettler	395/140
5,631,825	A		5/1997	van Weele et al.	345/339
5,761,429	A	*	6/1998	Thompson	709/224
5,793,974	A	*	8/1998	Messinger	709/224
5,801,702	A	*	9/1998	Dolan et al.	345/357
5,958,012	A	*	9/1999	Battat et al.	709/224
6,008,814	A	*	12/1999	Baldwin et al.	345/427

FOREIGN PATENT DOCUMENTS

EP	029121 A1	*	5/1981
EP	0 529 121 A1		3/1993
EP	0 609 030 A1		8/1994
EP	0 694 829 A1		1/1996
JP	0 4318684		10/1992

OTHER PUBLICATIONS

Harsha P. Kumar, Catherine Plaisant and Ben Shneiderman, Browsing Hierarchical Data with Multi-level Dynamic Queries and Pruning, International Journal of Human–Computer Studies, vol. 46, No. 1, XP002088976, pps. 103–124 (1997).

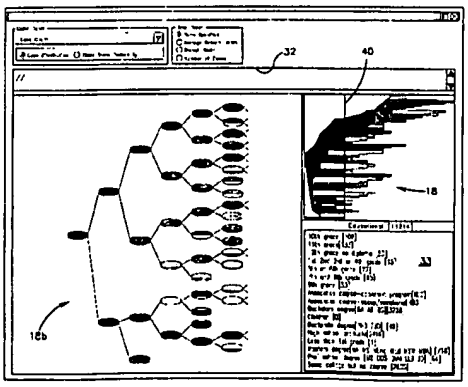
* cited by examiner

Primary Examiner—Kristine Kincaid
Assistant Examiner—Thomas T. Nguyen
(74) *Attorney, Agent, or Firm*—Watts, Hoffmann, Fisher & Heinke, Co., LPA

(57) **ABSTRACT**

Method and apparatus for displaying and navigating data organized in the form of a graph structure (hierarchy or network) is presented. The invention has application for displaying a system of interconnected nodes such as a graph, a network, an organizational chart, a flowchart etc. wherein data or information is associated with nodes of the system. A user interface is implemented as an ActiveX control having a viewer component for displaying and navigating graph structur (for example a data mining model over data records or a directory structure over a set of files). The viewer component updates the contents of related windows that display different aspects of the components (nodes) of the data structure. A thumbnail window presents the user with an overview of the data structure. A layout window presents a more detailed view of part of the graph structure. Other windows display context and detailed properties associated with particular selected nodes. One instance of the invention is used for displaying structure of a database classifier which organizes data in a tree. A tree viewer maintains a depiction of the entire graph (or tree) in the Thumbnail window and depicts a detailed portion of the graph in a larger layout window. The user can move the mouse pointer over either the thumbnail or the layout window and by mouse actuated inputs can control the manner in which the window depicts the tree structure. Color coding of properties of the structure being displayed, along with auxiliary detail windows for displaying values and histograms, can be used to quickly navigate a large structure and locates zones of interest within it.

4 Claims, 6 Drawing Sheets



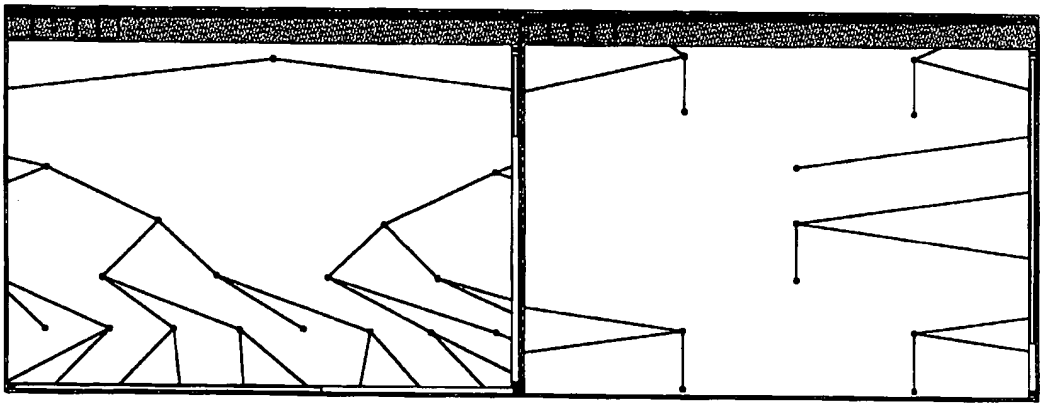


Fig.1
PRIOR ART

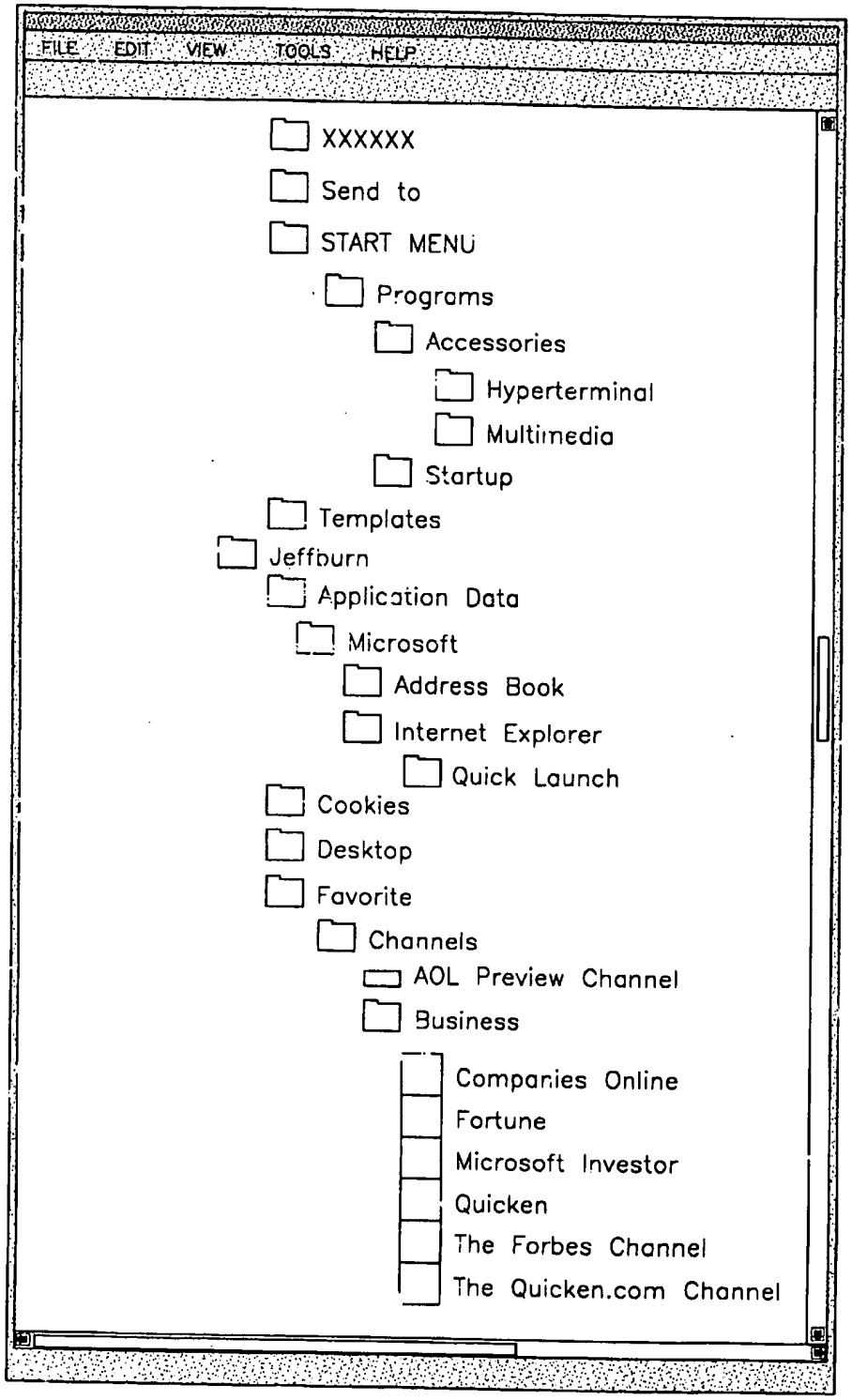


Fig.2
PRIOR ART

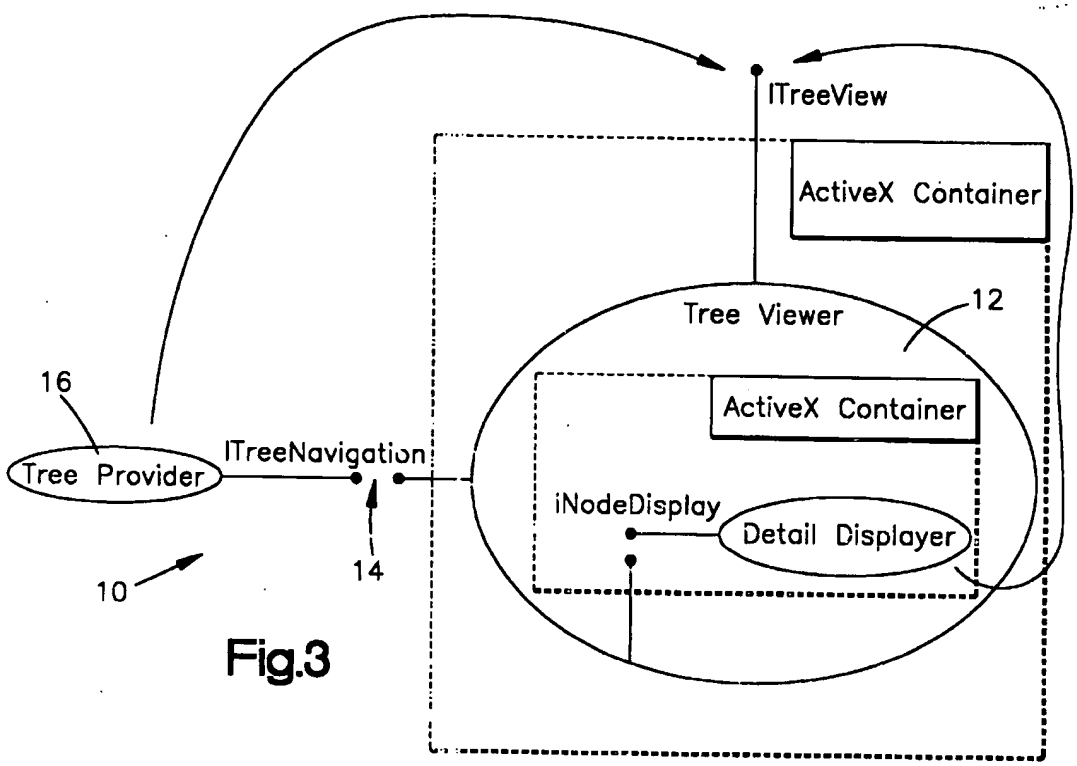

ITreeView
ActiveX Container
Tree Viewer
12
16
ActiveX Container
ITreeNavigation
Tree Provider
iNodeDisplay
Detail Displayer
14
10

Fig.3

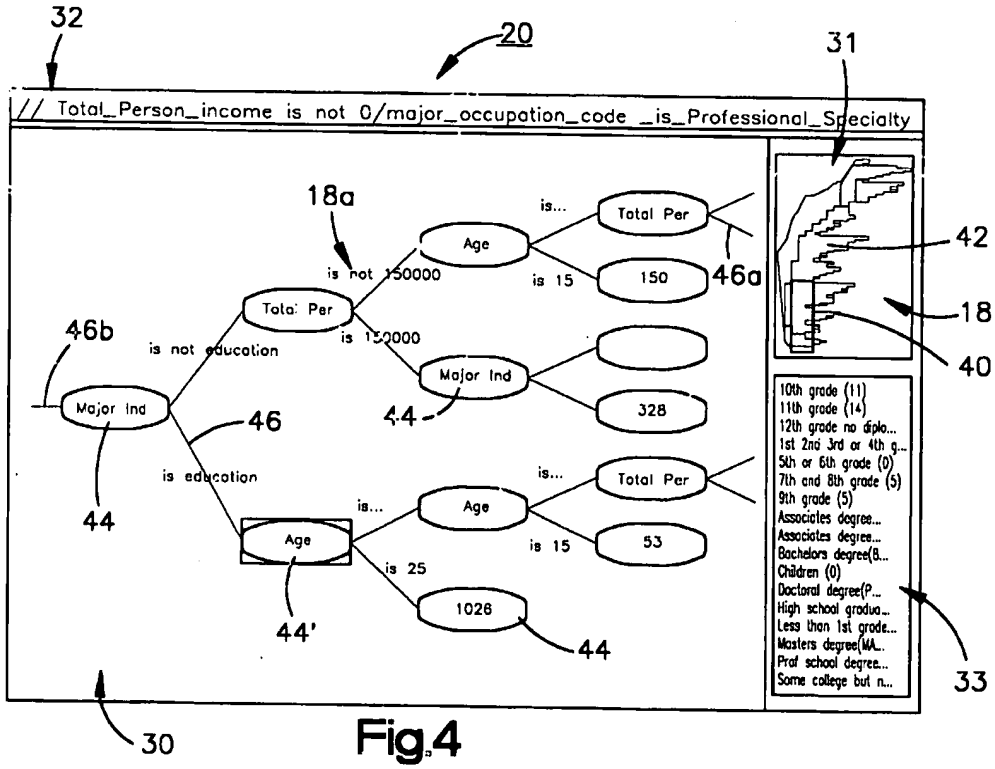

32
20
31
// Total_Person_income is not 0/major_occupation_code _is_Professional_Specialty
18a
Age
is...
Total Per
is not 150000
is 15
150
46a
Total Per
46b
is not education
is 150000
Major Ind
328
Major Ind
46
44
44
is education
is...
Total Per
is...
Age
Age
is 15
53
is 25
1026
44'
44
42
18
40
10th grade (11)
11th grade (14)
12th grade no diplo...
1st 2nd 3rd or 4th g...
5th or 6th grade (0)
7th and 8th grade (5)
9th grade (5)
Associates degree...
Associates degree...
Bachelors degree(B...
Children (0)
Doctoral degree(P...
High school gradua...
Less than 1st grade...
Masters degree(MA...
Prof school degree...
Some college but n...
33
30

Fig.4

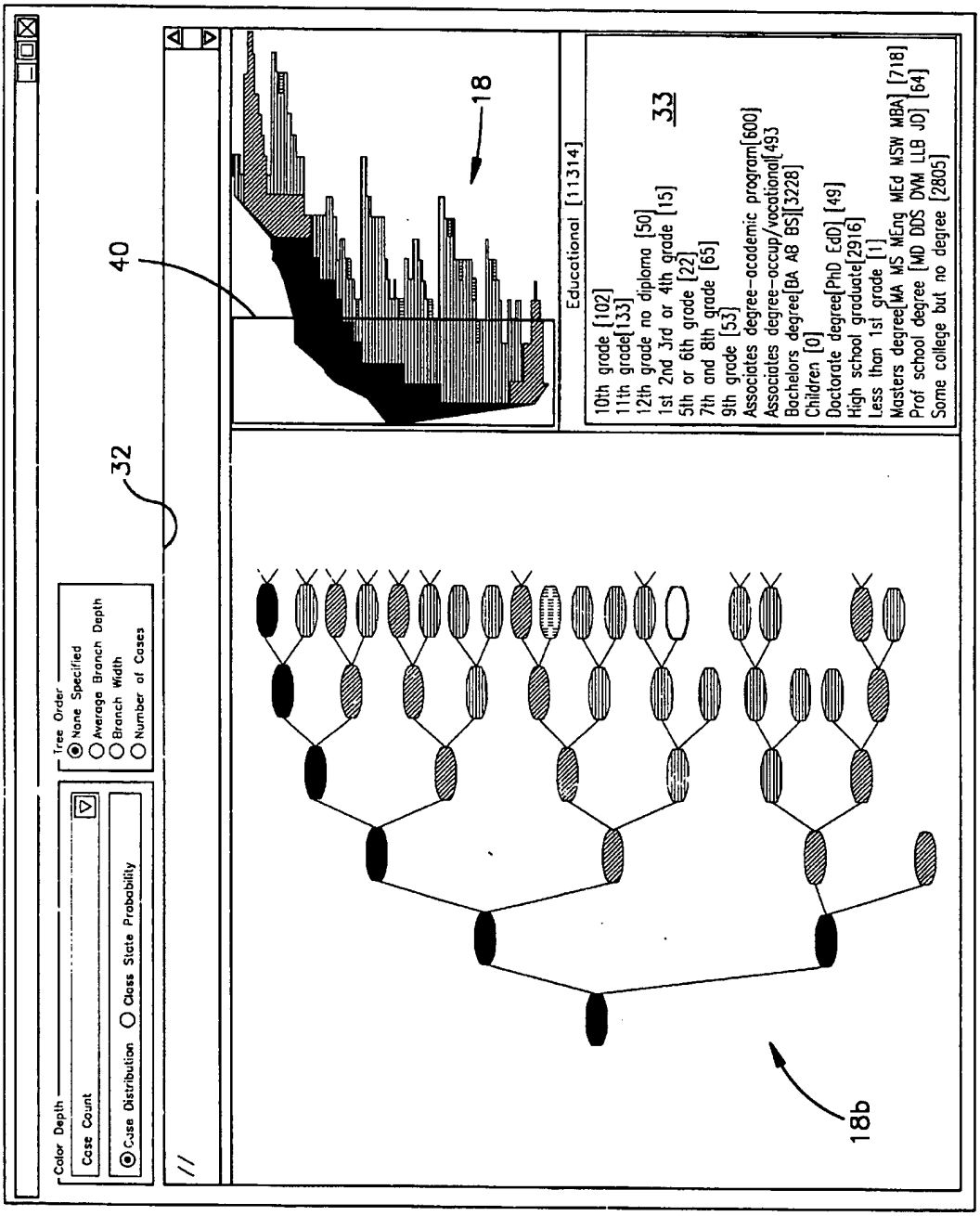

Color Depth
Case Count
Case Distribution
Class State Probability
Tree Order
None Specified
Average Branch Depth
Branch Width
Number of Cases
40
32
18
Educational [11314]
10th grade [102]
11th grade[133]
12th grade no diploma [50]
1st 2nd 3rd or 4th grade [15]
5th or 6th grade [22]
7th and 8th grade [65]
9th grade [53]
Associates degree-academic program[600]
Associates degree-occup/vocational[493
Bachelors degree[BA AB BS][3228]
Children [0]
Doctorate degree[PhD EdD] [49]
High school graduate[2916]
Less than 1st grade [1]
Masters degree[MA MS MEng MEd MSW MBA] [718]
Prof school degree [MD DDS DVM LLB JD] [64]
Some college but no degree [2805]
33
18b

Fig.5

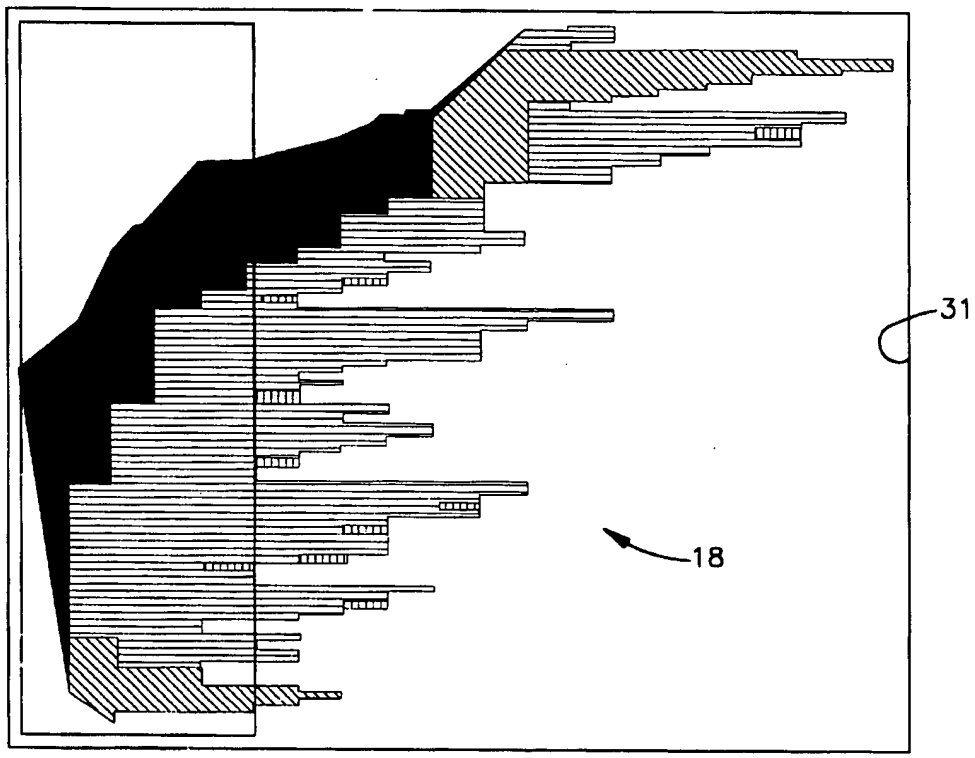


Fig.5A

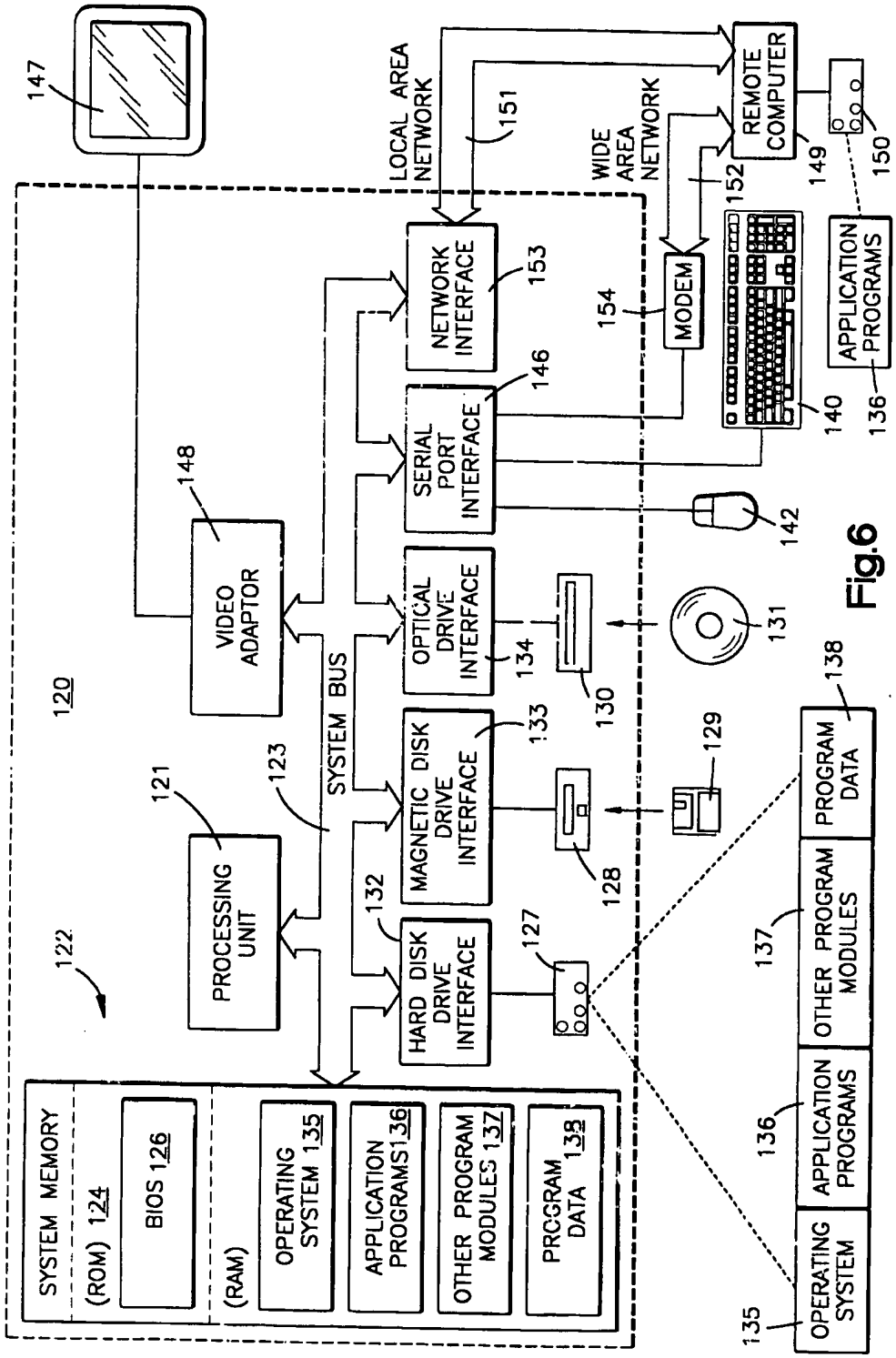

147
LOCAL AREA NETWORK
151
WIDE AREA NETWORK
REMOTE COMPUTER
149
150
152
NETWORK INTERFACE
153
154
MODEM
APPLICATION PROGRAMS
136
140
146
SERIAL PORT INTERFACE
148
142
VIDEO ADAPTOR
OPTICAL DRIVE INTERFACE
134
131
120
SYSTEM BUS
130
121
123
MAGNETIC DISK DRIVE INTERFACE
133
129
PROGRAM DATA
138
PROCESSING UNIT
132
128
122
HARD DISK DRIVE INTERFACE
127
OTHER PROGRAM MODULES
137
SYSTEM MEMORY
(ROM) 124
BIOS 126
(RAM)
OPERATING SYSTEM 135
APPLICATION PROGRAMS 136
OTHER PROGRAM MODULES 137
PROGRAM DATA 138
APPLICATION PROGRAMS
OPERATING SYSTEM
135

Fig.6

METHOD AND APPARATUS FOR VISUALIZING AND EXPLORING LARGE HIERARCHICAL STRUCTURES

CROSS REFERENCE TO RELATED APPLICATIONS

The present patent application claims priority of co-pending U.S. Provisional patent application serial No. 60/099,685 entitled "Method and COM Control for visualizing and Exploring Large Hierarchies of Trees of Information" which was filed in the United Sates Patent and Trademark Office on Sep. 10, 1998 pending.

FIELD OF THE INVENTION

The present invention concerns data visualization wherein structure has been imposed on data and a means of displaying the structure is needed. Traditional methods for displaying structures (such as hierarchies) are difficult for people to use when the structures get large.

BACKGROUND ART

Data reduction schemes such as those used in the mining of data from a large database impose a structure onto the data to better understand that data. Often a tree (or hierarchical) representation of the data is provided. A tree representation can often ease the viewing, accessing or understanding of the data represented by the tree. Tree structures are particularly convenient for separating large databases into segments or subsets of data. A set of files in a computer system are usually also represented as a hierarchy of directories with the leaves being individual files. This is also true for books in a library catalog system, and so forth. In general a tree has one top level or "root" node which can have two or more branches emanating from it. The branches represent some logical separation of the data. Each of these branches ends in another node, which can in turn have branches leaving from it, or the node can be a termination point or "leaf" of the tree (no more branches). Examples of data structured as a tree include the directory structure of a computer file system, a database table representing a "bill-of-materials" relationship, and the organization chart of a corporation. Examples from data mining include decision trees for classification and hierarchies of clusters (segments) generated from a hierarchical agglomerative clustering algorithm or a similar method.

Other examples of structures that are used to impose order to a large data set are networks or graphs. These structures do not have a single root but do have nodes that are interconnected by edges. Local area and wide area networks are examples of structures containing data which can conveniently depicted as a graph of nodes indicating for example nodes on a network. Such a graph could be used to indicate traffic on the network wherein data passing though a transmission node would be represented as data within a node of the graph.

In data mining, especially in building decision trees for prediction over a database, it is frequently the case that a very large tree is produced. An example of a decision tree for use in data mining is disclosed in copending U.S. patent application Ser. No. 08/982,760 entitled "Method and Apparatus for Efficient Mining Classification Models from Databases" to Chaudhuri et al which is assigned to the assignee of the present invention. Viewing an entire tree or browsing the data using the model extracted (the tree) is very challenging when the tree is large (has many nodes). Most prior art systems for displaying data structures such as trees display the tree and then zoom in and out to show either smaller or larger portions of the tree. These prior systems make it difficult to browse the tree structure in detail, while continuing to provide the user a context of what portion of the tree structure is being viewed.

Because of the hierarchical branching inherent in a typical tree structured data set, the "width" of the tree tends to increase exponentially with the "depth". For example, a balanced tree that has on average 4 branches per node will have 4 nodes at the 2^{nd} level of the hierarchy, 16 nodes at the 3^{rd}, 256 nodes at the 5, and 4^{n} at the nth level.

Traditional methods for displaying a tree in a user interface use an equally sized object for each node in the tree. The tree can be laid out graphically as a network of connected objects in a window with scroll bars. Another example of a prior art tree representation is the hierarchy of files and directories displayed by the Microsoft Windows Explorer program. In both of these examples provision must be made to collapse or expand a node in order to make the navigation of a large unwieldy tree manageable.

For certain situations, such a tree may need to be seen in its entirety (fully expanded). If the tree is scaled down so that it can be viewed completely at one time, then not much useful information can be shown along with the nodes of the tree. If the entire tree is laid out so that usable information can be shown on each node, then certain problems arise. When the top of the tree is viewed, the distance between the high level nodes can become so great that they are of no use. When the bottom of the tree is put into view, the lower level nodes become a tangle of seemingly disconnected information (it is difficult to see the relationships between the nodes because connections to the parent nodes cannot be seen).

FIGS. 1 and 2 illustrate these problems. FIG. 1 depicts different visual views of a large amount of data in the form of a tree. One view depicts data nodes near a top (left side in FIG. 1) and a second view depicts a different set of data nodes near a base or bottom (right side of FIG. 1) of the data tree. A scroll bars are used to navigate the tree structure that is displayed in the scrollable window. FIG. 2 is an example of a fully expanded directory structure depicted by Windows Explorer where all connections to the higher level, owner directories have been lost.

SUMMARY OF THE INVENTION

The present invention concerns a method for enabling effective browsing and examination of large amounts of data that are organized or classified in a data structure. Many of the problems that have been experienced trying to explore and/or view large amounts of data are overcome by a novel navigation and rendering scheme constructed in accordance with the invention.

Two simultaneously viewable windows are displayed for a user. Using the example of a data tree, an overview of the entire tree is depicted in one window and only a portion of the tree is displayed in a second window. The second window shows individual nodes and interconnections and the first overview window depicts the entire tree in a way that makes traversal of the information in the tree intuitive to the user.

One use of the invention is for viewing a decision tree produced by a data mining system such as the data mining system disclosed in co-pending U.S. patent application Ser. No. 08/982,760 entitled "Method and Apparatus for Efficiently Mining Classification Models from Databases." Another representative use of the invention displays and

navigates a file structure maintained by a computer operating system. Generally, the invention has application for displaying and system of interconnected nodes such as a graph, a network, an organizational chart, a flowchart etc. wherein data or information is associated with nodes of the system.

Use of color gradients helps the user identify trends or anomalies in the data by visualizing the tree as a whole. In general, a property is associated with a clor, and the color intensity can visually represent the value of the property (e.g. red being high, yellow being low). An exemplary embodiment of the invention is implemented as an ActiveX control with a user interface suitable for viewing and exploring large trees.

One exemplary embodiment of the invention includes a method for displaying data as a tree data structure. A user interface is painted by a tree rendering component that allows intuitive navigation and interpretation of the tree structure. The tree rendering component updates two related windows, a layout window and a thumbnail window. The tree rendering component maintains a structure of a tree depicted in the thumbnail window and depicts a portion of the entire tree in the layout window. The use of side by side windows, one of which shows the whole tree and another of which shows a portion of the tree allows easier user visualization of the data characterized by the tree.

The exemplary embodiment of the invention also conveys additional information in other windows (or window panes of the main window) on a viewing monitor. In accordance with one embodiment, a path window displays as text a sequence of concatenated decision steps required to reach a given node in the data structure. It is a textual summary of the context. Additionally, the user can select a given node that is displayed in the layout window and a detail window itemizes information about the contents of the selected node. The detail window can include a histogram of the values of a variable or score of interest. In use of the invention with a database classifier such a window could further itemize different categories of data that satisfy the logic leading to a particular node of the data structure.

These and other objects, advantages and features of the invention are further understood from the detailed description of an exemplary embodiment of the invention which is described in conjunction with the accompanying drawings.

BRIEF DESCRIPTION OF THE DRAWINGS

FIGS. **1** and **2** are prior art illustrations of data depictions;

FIG. **3** is a high level architecture of a data displaying system constructed in accordance with the present invention;

FIGS. **4** and **5** are alternate depictions illustrating different aspects of the invention for use in displaying data structures;

FIG. **5A** is a enlarged view of a thumbnail window for displaying a data structure in an overview fashion; and

FIG. **6** is a schematic of a computer system for implementing the exemplary embodiment of the invention.

DETAILED DESCRIPTION OF AN EXEMPLARY EMBODIMENT OF THE PRESENT INVENTION

An exemplary embodiment of the present Invention is implemented as an ActiveX computer control component **10** that includes a data viewer component **12** for displaying data structures on a viewing screen such as a computer screen **147** by supplying appropriate data for use by the comput-computer operating system that updates the screen display. The techniques for implementing Active X components by means of COM object interfaces are known in the prior art. A discussion of COM object interfaces is contained in the book "Inside COM" by Dale Rogerson, Microsoft Press copyright 1997. Although the data viewer component disclosed in the application displays data as a tree structure it is appreciated that other data depictions such as graphs and networks could be displayed using the Active X components **10**. In the discussion that follows, data depiction and data structure are used interchangeably.

Practice of the invention allows a user to better visualize and navigate a data structure derived from a data source such as a large database stored on multiple (possibly distributed) memory devices. A user Interface **20** (FIG. **4**) shows an example of operation of the data viewer, i.e. a display on a viewing screen such as a computer monitor **147** (FIG. **6**). Within a frame window, the ActiveX implementation of the viewer sets up four related windows **30–33** to help the user navigate the data structure from the data source. These four windows **30–33** are identified in FIG. **4** as a layout window **30**, a thumbnail window **31**, a path window **32**, and a details window **33**. The data viewer component **12** is coupled by means of an ITreeNavigation COM interface **14** to a tree provider component **16**. The tree provider **16** maintains the structure of a tree **18** depicted in the layout window **30** and provides all contextual information about the tree. By responding to some simple generic requests from the tree viewer **12**, the tree provider **16** draws and navigates the tree. The tree viewer component **12** is unconcerned with the type or meaning to the tree being drawn. The tree provider **16** also supplies a Detail Displayer ActiveX control for displaying detailed provider specific information about individual nodes within a tree. FIG. **3** shows the high level architecture of how the tree viewer **12** interfaces with the tree provider **16** as a COM interface **14**. The results of the cooperation between the tree provider and the tree viewer is an ItreeView interface to the operating system.

The present invention has particular utility for use in characterizing data contained in a database having many records stored on multiple, possibly distributed storage devices. Each record has many attributes or fields which for a representative database might include age, income, number of children education level, marital status etc. Such data can be obtained, for example, from census data gathered from many people in response to a survey. One goal of the invention is to help a user to visualize the contents of the database once it has been classified by means of a classifier scheme such as the scheme disclosed in the above mentioned patent application to Chaudhuri et al which is incorporated herein by reference.

Thumbnail to Layout Relationship

The display and operation of the Thumbnail window **31** is closely tied to the Layout window **30**. The relationship between the two is similar to the movement co-ordination between the display in a client window in response to a user actuation of a scrollbar. The Thumbnail view shows a miniature outline of an entire tree **18**. The Layout view shows a full size representation of a portion **18*a*** of the tree (a sub-tree). A raised area control component **40** within the Thumbnail view acts much like the "thumb" of a normal scollbar and will hereinafter referred to as the thumb **40**. The position of this thumb **40** over the miniature view of the tree **18** determines the portion **18*a*** of the tree (sub-tree) that is shown in the Layout view. Under most circumstances, the sub-tree under the thumb is completely visible within the Layout window because the size of the thumb is determined by the largest sub-tree that can be viewed in the layout window **30**.

Navigation with the Thumbnail

The Thumbnail view's rendering includes a raised control component or thumb **40** which can be moved to a specific region of the miniature tree **18** simply by moving the cursor over a portion of the tree and clicking on another portion **42** of the tree. Additionally, the thumb **40** can be moved incrementally by clicking on the thumb with the pointing device. While the cursor is positioned over the thumb, the cursor (not shown) changes shape to indicate the direction of incremental movement that a user actuated click of the mouse or other pointing device will produce. As the raised area control component **40** is moved in this manner, its size is changed to indicate the set of nodes in the subtree that can be fit in the main layout window. It is appreciated that other bound shapes of indicators could be used to give a relative indication of data displayed in the layout window to the entire tree structure.

Drawing the Layout Window

The sub-tree **18***a* selected by the raised area of the Thumbnail view is drawn at a larger scale as a Layout view in the layout window **30**. The sub-tree is drawn left to right where the leftmost node represents the highest point in the tree outlined by or encompassed by the borders of the thumb **40** of the thumbnail view **18**. (This node is a common parent of all other visible nodes depicted in the layout view.) In the figure the tree is laid out in left to right order to include descritptive text naturally. The tree may be laid out in any other fashion by the tree rendering component.

Nodes are drawn as colored ellipses **44** with branches shown as lines **46** leaving the right side of a node and connecting to the node's children. The color of the node (as well as the corresponding region of the Thumbnail) is determined by a score that is reported for the node by the tree provider **16**. The score can be changed via some user interface controls, and the tree provider accordingly can expose different scores for each node. A text label is draw within the ellipse of the node, and a text label for an edge is drawn on any connecting line. The tree provider component **16** supplies these labels.

As is well known, a standard video display monitor is wider than it is tall. Tree structures having interconnected nodes are best displayed on such a monitor if the tree is displayed 'sideways'. If the leftmost node is the tree root, then no edge leads to the node on the left hand side of the display. If the leftmost node is an interior node, then an edge is connected to the leftmost node. The text that is rendered within the layout window for conveying information about the tree nodes is also displayed horizontally next to the nodes and edges of the tree.

The visible sub-tree **18***a* is drawn within the Layout window as if it was a complete tree. Children nodes that are not included within the sub-tree and will not be visible are not considered when spacing the rightmost visible nodes. The Layout view draws the visible sub-tree to a given scale. The window scale can be changed providing a zoom-in or zoom-out effect.

Navigation within the Layout Window

The tree can also be navigated from within the Layout view. Clicking on an edge line **46** will change the display to show a sub-tree starting with the child node that the edge line points to. This includes any half-lines **46***a* that point toward non-visible children. (lines on the right of the rightmost nodes.) One exception is that clicking on an edge **46***b* emanating from the left of the leftmost (top) node will move the display up one level in the tree.

One of the nodes within the Layout view can be put into focus. A node in focus is drawn differently to indicate selection.

Details of Layout/Thumbnail Interaction

The Thumbnail and Layout windows work together to determine the size of the Thumbnail's scroll thumb and thus the size of the sub-tree visible within the Layout view. The following steps describe this interaction: A thumbnail component of the tree viewer **12** is instructed (by a navigation or a COM interface call) to put a sub-tree starting at a particular node into view. The Thumbnail asks the Layout for the dimensions of the largest possible tree that it can fit without clipping inside its window. These numbers are based upon the dimensions of the Layout window, the current Layout scaling factor, and some inherent metrics about the relative sizes and relative positions of the object used to render a tree.

The Thumbnail control then knows the maximum depth or width that can be shown, and traverses down the tree starting from the chosen node. It keeps track of the depth traversed and the width of the sub-tree at the given depth and stops traversing down before one of the maximums is surpassed. The Thumbnail now can identify the visible sub-tree, and it positions its thumb (raised region) accordingly on the miniature tree **18**. The Layout view gets the depth of the visible sub-tree **18***a* from the Thumbnail and draws the tree.

The Path Window

The Path window **32** shows a concatenation of the decision criteria needed to reach the Layout window's leftmost visible node by working down from the root of the entire tree. The tree provider **16** supplies a description of each step along the way. The Path provided in this window gives the user a context when viewing a nested sub-tree **18***a* since it gives information about the leftmost data node of that subtree.

The Details Window or Windows

When one of the nodes is selected for focus within the Layout view, the Details window can show detailed provider specific information about the node. To make this happen, the tree provider **16** supplies an ActiveX control that supports the INodeDisplay interface of a component for displaying node detail. This provider-supplied control will be hosted within the Details window. The tree viewer uses the INodeDisplay interface to alert a provider object that the details for a specific node are to be shown when a particular node is in focus. In FIG. **4** the node **44'** is in focus. This node was selected by the user moving the cursor over the node and right clicking on the mouse **142** (FIG. **6**) or other suitable pointing device.

Visualizing Tree Structure via Color Gradients

The tree provider **16** has control over the semantics for the "score" that it reports for a node to the tree viewer. This score (between 0 and 1) is used by the tree viewer to set the illumination of the color of each node's representation in the Layout view and the miniature tree outline for nodes in the thumbnail window. The FIG. **5**, the tree viewer illustrates a decision tree that classifies education level for a large sample of census data. The score reported for nodes shown in FIG. **5** is relative to the number of data records for individuals that conform to or fit a given edge's condition. Using this score, the tree shows dark "veins" of data concentration since a score is based upon the number of records relative to the entire database represented by a node and the children of that node.

FIG. **5** and FIG. **5A** illustrate by means of cross hatching in the drawings use of the color coding to depict data structure information. The control component **40** is located over the leftmost portion of the tree **18**. As seen in the path window **32** the entire tree is chosen so that the leftmost node

in the window **30** is the tree root node and has no edge entering the node from the left. Each of the nodes of the window **30** has been color coded in accordance with a scale factor based on the number of data records from the database classified in a given node. Branches in the tree are based on attribute values of the data records in the database from which the records are obtained and therefore as one travels away from the root node in the tree the score or number of records decreases. In the depiction of FIGS. **5** and **5A** the largest number of records is indicated with a 'black' cross hatching, the second highest with a 'green', then a 'blue' a 'violet' and finally a 'white'.

The score for the node as determined by the tree viewer need not be based on an absolute number but instead can be chosen as a relative indication. In the tree classifier of the previously identified patent application to Chaudhuri et al, for example, the nodes of the tree correspond to a classification system that can result in records having the same attribute satisfying different branches of the classifier tree. The color of the nodes could be used to indicate therefore a probability of a given node satisfying certain facts rather than an absolute tabulation of a number of data records. Using an example from a database classifier, the color coding could signify probability of a family having a child in college even though the branching of the tree does not branch on that attribute value.

Other properties could be color coded to convey information about the data structure. Instead of a classifier of a database, the tree structure could depict a directory structure of a computer file system. In that circumstance, in addition to showing the size of the files within the directory, a use somewhat akin to the one described above for a database classifier, the color coding provided by the Tree provider could indicate how recently files in the directory structure have been opened, frequency of use, permission levels of files with the file structure or any other property that could be turned into a score relative to other nodes in the hierarchy of files. Similarly, in a graph structure or network the color coding could indicate data traffic or instances of data retransmission or error occurrences at a particular node of the network.

Computer System

With reference to FIG. **1** an exemplary data processing system for practicing the disclosed invention includes a general purpose computing device in the form of a conventional computer **120**, including one or more processing units **121**, a system memory **122**, and a system bus **123** that couples various system components including the system memory to the processing unit **121**. The system bus **123** may be any of several types of bus structures including a memory bus or memory controller, a peripheral bus, and a local bus using any of a variety of bus architectures.

The system memory includes read only memory (ROM) **124** and random access memory (RAM) **125**. A basic input/output system **126** (BIOS), containing the basic routines that helps to transfer information between elements within the computer **120**, such as during start-up, is stored in ROM **24**.

The computer **120** further includes a hard disk drive **127** for reading from and writing to a hard disk, not shown, a magnetic disk drive **128** for reading from or writing to a removable magnetic disk **129**, and an optical disk drive **130** for reading from or writing to a removable optical disk **131** such as a CD ROM or other optical media. The hard disk drive **127**, magnetic disk drive **128**, and optical disk drive **130** are connected to the system bus **123** by a hard disk drive interface **132**, a magnetic disk drive interface **133**, and an optical drive interface **134**, respectively. The drives and their associated computer-readable media provide nonvolatile storage of computer readable instructions, data structures, program modules and other data for the computer **120**. Although the exemplary environment described herein employs a hard disk, a removable magnetic disk **129** and a removable optical disk **131**, it should be appreciated by those skilled in the art that other types of computer readable media which can store data that is accessible by a computer, such as magnetic cassettes, flash memory cards, digital video disks, Bernoulli cartridges, random access memories (RAMs), read only memories (ROM), and the like, may also be used in the exemplary operating environment.

A number of program modules may be stored on the hard disk, magnetic disk **129**, optical disk **131**, ROM **124** or RAM **125**, including an operating system **135**, one or more application programs **136**, other program modules **137**, and program data **138**. A user may enter commands and information into the computer **120** through input devices such as a keyboard **140** and pointing device **142**. Other input devices (not shown) may include a microphone, joystick, game pad, satellite dish, scanner, or the like. These and other input devices are often connected to the processing unit **121** through a serial port interface **146** that is coupled to the system bus, but may be connected by other interfaces, such as a parallel port, game port or a universal serial bus (USB). A monitor **147** or other type of display device is also connected to the system bus **23** via an interface, such as a video adapter **148**. In addition to the monitor, personal computers typically include other peripheral output devices (not shown), such as speakers and printers.

The computer **120** may operate in a networked environment using logical connections to one or more remote computers, such as a remote computer **149**. The remote computer **149** may be another personal computer, a server, a router, a network PC, a peer device or other common network node, and typically includes many or all of the elements described above relative to the computer **120**, although only a memory storage device **150** has been illustrated in FIG. **1**. The logical connections depicted in FIG. **1** include a local area network (LAN) **151** and a wide area network (WAN) **152**. Such networking environments are commonplace in offices, enterprise-wide computer networks, intranets and the Internet.

When used in a LAN networking environment, the computer **120** is connected to the local network **151** through a network interface or adapter **153**. When used in a WAN networking environment, the computer **120** typically includes a modem **154** or other means for establishing communications over the wide area network **152**, such as the Internet. The modem **154**, which may be internal or external, is connected to the system bus **123** via the serial port interface **46**. In a networked environment, program modules depicted relative to the computer **120**, or portions thereof, may be stored in the remote memory storage device. It will be appreciated that the network connections shown are exemplary and other means of establishing a communications link between the computers may be used.

While the present invention has been described with a degree of particularity, it is the intent that the invention include all modifications and alterations from the disclosed implementations falling within the spirit or scope of the appended claims.

What is claimed is:

1. Apparatus for displaying a data structure which is based on information contained within a data source comprising:

a data classifier for classifying the information from the data source and building a data structure from the information;

a display component communicating with the data classifier for rendering a visual depiction of the data structure by displaying two related renderings of the data structure, a first overview depiction of the data structure and a second detail depiction of a subportion of the overview portion wherein the detail depiction is displayed as a plurality of interconnected nodes that convey information about a data set that defines an associated node; said display component also defining a highlighted region of the overview depiction and depicting correspondingly different portions of the data structure in the detail depiction based on the highlighted region; and

a control for allowing a user to communicate with the display component to adjust the highlighted region and thereby adjust the visual rendering of the data structure; and

means for displaying a third related rendering comprising a textual description of the nodes that make up the data structure;

said display component adjusting a size of the highlighted region based upon the data contained within the portion of the data structure represented by the highlighted region.

2. The apparatus of claim 1 additionally comprising means for depicting additional text information for a node that is highlighted in the detail depiction.

3. The apparatus of claim 2 wherein the display renders a detail window for displaying the detail depiction and separately renders an additional detail window containing information concerning nodes in the detail window.

4. A method for displaying data comprising the steps of providing a user interface for displaying and navigating a data structure characterized by a decision tree representing data from a database by:

rendering an overview depiction of a first portion of the decision tree in one region of a video display;

rendering a second, detail depiction of the decision tree in a second region of the video display wherein the detail depiction of the decision tree is displayed as a plurality of interconnected nodes from a selected portion of the decision tree that convey information about a data set from the database;

displaying a control component within the overview depiction for navigating the decision tree; changing a position of said control component with respect to the overview depiction; adjusting the detail portion of the decision tree appearing within the second region of the video display based on a position of the control component; and changing the size of the control component based on the detail portion of the data structure appearing within said second region; and

rendering in a third region of a video display text corresponding to a logic of the branches of the decision tree leading to the data nodes displayed in the detailed depiction.

* * * * *